MICROCOMPUTADOR CURSO PRÁTICO

TIRE O MÁXIMO DO MICRO EM 30 SEMANAS

PUBLICAÇÃO SEMANAL ILUSTRADA

19

Cr$ 10.250

Osborne Encore • Código de máquina
Revolução musical com a MIDI

rioGráfica

MANAUS, ALTAMIRA, MACAPÁ, JI.PARANÁ, PORTO VELHO, RIO BRANCO, SANTARÉM, BOA VISTA, SINOP, ALTA FLORESTA: Cr$ 13.330

MICROCOMPUTADOR CURSO PRÁTICO

CHIPS & BYTES

Tal e qual 377

PROJETOS DE PROGRAMAS

Controle auditivo 379
Relações produtivas 388
Tentativa e erro 390

SOFTWARE

Criação de registros 380

APLICAÇÕES

O protocolo MIDI 382

RAIO X

Osborne Encore 385

LINGUAGENS

Introdução ao ASSEMBLY 392

PERFIL

Psion 396

TESTE

Depois de ler com atenção todo o fascículo, responda a estas perguntas. Veja a solução no próximo número, junto com um novo teste.

1) Os bancos de dados comuns, do tipo tabela, são relacionais, hierárquicos ou em rede?
2) Em termos de economia de memória, qual dos sistemas de banco de dados é o mais eficiente?
3) A interface MIDI para sintetizadores musicais opera com transmissão serial ou paralela?
4) Qual é a via de comunicação de uma telecopiadora?

RESPOSTAS DO TESTE ANTERIOR

1) "Tartaruga" é o nome genérico dado a qualquer dispositivo do tipo robô dirigido por comandos em linguagem LOGO ou similar.
2) O dispositivo digital que coordena diversas linhas melódicas e determina sua duração é o seqüenciador.
3) Existe uma versão do processador de textos WordStar, disponível para micros de 16 bits, que mostra na tela todos os caracteres da língua portuguesa tal como são escritos.
4) SGBD significa Sistema Gerenciador de Banco de Dados.

PLANO DA OBRA

MICROCOMPUTADOR - CURSO PRÁTICO é uma coleção de 30 fascículos de periodicidade semanal. Os 29 primeiros terão, cada um, vinte páginas internas (miolo) e quatro capas, além de uma introdução de dezesseis páginas acompanhando o fascículo 1. O fascículo 30 incluirá o índice geral da obra. Os miolos, encadernados, constituem dois volumes de um curso prático de microcomputação, com seções indicadas por tarjas de diferentes cores.
As duas primeiras capas são descartáveis; as 3.as e 4.as trazem programas de jogos. Encadernadas, formam um volume de cem páginas.

COMO ENCADERNAR

As capas duras, incluindo as guardas, estarão à venda simultaneamente com os fascículos 10 (**Volume 1**) e 30 (**Volume 2** e **Jogos**).
Volume 1 - Deve ser encadernado com os elementos dispostos nesta ordem: guardas; frontispício e introdução (publicados com o fascículo 1); miolos dos fascículos de 1 a 15; e guardas.
Volume 2 - Guardas; frontispício (que virá com o fascículo 30); miolos dos fascículos de 16 a 30; e guardas.
Jogos - Guardas; frontispício e sumário (que constituem as 3.as e 4.as capas do fascículo 30); 3.as e 4.as capas dos fascículos de 1 a 29; e guardas.

rioGráfica **CONSELHO DE ADMINISTRAÇÃO:** Roberto Irineu Marinho; João Roberto Marinho; José Roberto Marinho; Oscar Neves; **Diretoria:** Oscar Neves; Danilo Esteves Costa; João J. Noro; Marcos March; Nilo Sérgio de Almeida; **GRUPO EDUCAÇÃO E CULTURA - Divisão Comercial - Diretor:** Jaime Rodrigues; **Gerente de Planejamento:** João de Azeredo Coutinho; **Gerente de Vendas:** Rubens Barbosa; **Coordenadores de Promoção:** Paulo Cesar M. Seixas; Edgard Mello M. Neto; **Gerente de Circulação:** Norberto Martin; **Supervisor de Operações:** Abel Pereira Pinto; **Editorial - Diretor:** João J. Noro; **Editor-chefe:** Maurício Rittner; **Administração Editorial:** Heitor de Souza Paixão; **Editor:** Jesse Navarro; **Editor-assistente:** Claudio Marcondes; **Revisores:** Isis Augusta Loyolla; Cacilda Guerra; Luiz Vicente Vieira Filho; Marcos Domingos Agathão; Maria da Graça Mendonça Couto; **Diretor de Arte:** Adalberto Cornavaca; **Chefe de Arte:** Bonifácio Duardes Miranda; **Chefe de Estúdio:** José Yuji Kuribayashi; **Assistentes de Arte:** Miguel Luis Escámez Simón; Edemilson Ribeiro da Silva; **Colaboradores - Editores-assistentes:** Isa Mara Lando; Luiz Carlos Pizarro Marin; **Texto:** Carlos Eduardo Silveira Matos; Jussara Maturo; Paulo Brito; **Preparação de Texto:** Luiz Carlos Cardoso; **Tradução:** Luiz Roberto de Godoy Vidal; Maria Clara Cescato; Maria Ester Menezes Martinho; Fernando Dorfman Knijnik; Newton Roberval Eichemberg; Regina Amarante; **Pesquisa:** Stella Maria Quentel; **Assessoria Técnica:** Geraldo Coen; Gustavo José Ferreira Grubba; Lauro de Lauro; Pierluigi Piazzi; Sérgio Jorge Batarce; Lucas Kenishi Shirahata. As fotos não creditadas pertencem à obra original. Segue-se o crédito das demais: Itautec. © APSIF Copenhagen, 1984; © Orbis Publishing Ltd., 1984; © Editora Rio Gráfica Ltda., 1985, para a língua portuguesa. **Composição:** Editora Rio Gráfica Ltda. e AM Produções Gráficas Ltda. **Impressão:** JBIG.

NÚMEROS ATRASADOS - Você poderá comprar os exemplares que faltam em sua coleção, pelo preço do último fascículo posto à venda, em sua banca de jornal preferida. Caso não consiga obtê-los, faça sua solicitação por carta endereçada diretamente à Editora Rio Gráfica Ltda., rua Itapiru, 1209, Rio Comprido, Rio de Janeiro, CEP 20251. O atendimento será feito por via postal. Nas cidades do Rio de Janeiro e de São Paulo, as compras poderão ser efetuadas pessoalmente nos seguintes endereços: Rio de Janeiro - Rua Itapiru, 1209, Rio Comprido; São Paulo - Rua Frei Caneca, 1152, Consolação.

Distribuidor exclusivo para todo o Brasil: Fernando Chinaglia - Rua Teodoro da Silva, 907, Rio de Janeiro. Distribuidor para Portugal: Electroliber Lda. - Rua Prof. Reinaldo dos Santos, 1488, Lisboa.

TAL E QUAL

A transmissão de documentos via telefone difundiu-se no Brasil graças à IFAX 3021, um equipamento fac-símile produzido pela Itautec com base em tecnologia desenvolvida pela Canon japonesa.

Em 1979, a Xerox lançou no mercado americano sua telecopiadora 485, capaz de reproduzir em menos de 1 minuto uma carta comercial padrão enviada à distância. Totalmente automatizada e mais veloz que suas precursoras, era o quarto modelo de equipamento fac-símile lançado no mercado pela empresa.

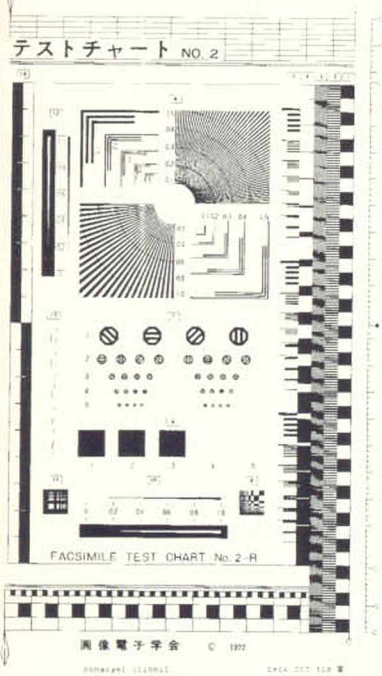
Tida como introdutora do sistema fac-símile também no Brasil, devido à comercialização do modelo Telecópia 400, a Xerox dividia o mercado com a Ecodata (subsidiária da Cable and Wireless, da Inglaterra) e a NEC do Brasil (subsidiária da Nippon Electronic Corporation, do Japão). No entanto, em função do alto custo e da falta de interesse pelo produto, esse equipamento importado não obteve sucesso.

O princípio do funcionamento da telecopiadora fac-símile é o mesmo dos equipamentos de telefoto e radiofoto: transmite por impulsos sonoros via rádio ou telefone.

A Itautec (Itaú Tecnologia), acreditando nas vantagens das fac-símile em relação ao tradicional sistema de transmissão via telex, enviou sete engenheiros de seu Departamento de Projetos ao Japão para um estágio de dois meses na Canon Corporation. Essa indústria detém a tecnologia do modelo FAX 510, que passou a ser produzido pela Itautec como IFAX 3021.

O modelo escolhido para comercialização no Brasil apresenta a característica da compatibilidade com outras telecopiadoras fac-símile, dos grupos 3 e 2, e com as mais antigas, baseadas no modo de transmissão FM (americano e euro-

Nacionalização crescente
O projeto de progressiva nacionalização da IFAX 3021 foi aprovado pela SEI (Secretaria Especial de Informática) em outubro de 1984. Já no mês seguinte, o equipamento era apresentado ao público durante a Feira de Informática, no Rio de Janeiro.



Qualidade das imagens
Nesta folha de teste, vêem-se os diversos recursos de resolução gráfica da fac-símile. Textos, mapas, fórmulas e diagramas podem ser transmitidos com todos os detalhes.

peu). A classificação em grupos (segundo o tempo de transmissão de uma página tamanho A4) se deve à padronização pelo CCITT (Comitê Consultivo Internacional de Telegrafia e Telefonia), com sede em Genebra, Suíça. No grupo 1, o tempo de transmissão vai a 6 minutos; no 2, equivale a 3 minutos; e, no 3, fica entre 15 segundos e 1 minuto. A IFAX 3021, portanto, é um equipamento de alta velocidade do grupo 3, mas opera também no grupo 2. Selecionando automaticamente a velocidade apropriada à transmissão, ajusta-a ao aparelho receptor. Cada transmissão realizada registra o modo como se emitiu (G2 ou G3), o número do telefone, o código dos assinantes (receptor e transmissor) e o número de páginas enviadas.

Em caso de erro de transmissão, o painel emite um sinal luminoso. A IFAX imprime o tipo de erro cometido e o número da página em que ocorreu. Isso permite rápida localização e correção do problema.

A recepção automática 24 horas por dia, opcional, permite que um documento seja recebido mesmo na ausência do operador. Além da recepção automática, a IFAX 3021 conta com um programa de transmissão de documentos, que apenas são colocados na bandeja, ficando dispensada a presença do operador. O alimentador de papéis permite posicionar até cinco originais na bandeja de documentos, transmitindo-os automaticamente.

A IFAX 3021 tanto pode imprimir cópias para checar a qualidade da imagem, antes da transmissão, como para uso local. Há três opções de contrastes, sendo que, para reproduções mais detalhadas, há ainda o modo fino, que possibilita maior resolução de imagem, para legibilidade máxima em qualquer dos contrastes. Durante a recepção ou transmissão, pode-se acionar a tecla [Falar], alertando o outro lado para atender o telefone antes ou depois de finalizada a operação.

O papel, do tipo termossensível, vem acondicionado em bobinas de 100 m, não exigindo trocas constantes; e a máquina se encarrega de cortá-lo no tamanho apropriado. A largura da imagem é de até 280 mm.

Na maioria das fac-símile, os documentos impressos são recebidos em ordem decrescente, o que requer posterior ordenamento à mão. A IFAX 3021 posiciona as cópias recebidas na ordem correta, dispensando esse trabalho.

A instalação requer uma tomada de 110 V e um telefone. Completada a ligação telefônica para onde se vai transmitir o documento, ouve-se um sinal telefônico contínuo; em seguida, o operador digita a tecla [Início] para acionar o processamento.

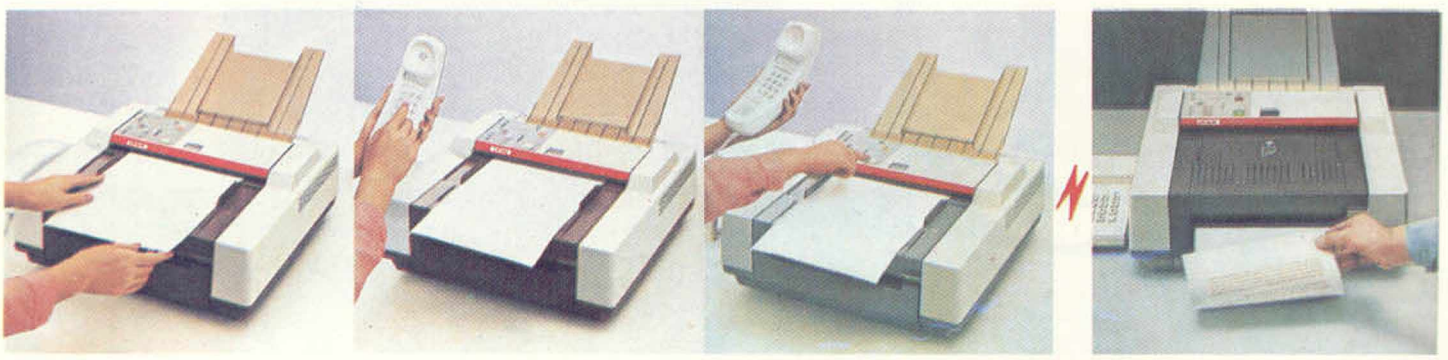
A fac-símile transmite qualquer tipo de original, manuscrito ou datilografado, impressões digitais, recortes de jornal, páginas de livro e impressos em qualquer tipo de papel. Atende a todo segmento de mercado que depende de comunicação rápida de textos, diagramas, gráficos, figuras, tabelas, fórmulas e mapas. Uma vez que o equipamento opera em branco e preto, sem o uso de meio-tom (cinza), a reprodução de fotos se faz em alto contraste. A velocidade de transmissão varia de 2.400 a 9.600 bauds (bits por segundo).

O sistema de leitura se faz por um chip que transforma sinais ópticos em elétricos. A leitura e o armazenamento se fazem linha a linha, sucessivamente. Como o papel de impressão vem enrolado, a transmissão-recepção é contínua. O sensor de imagem divide a linha da varredura horizontal em elementos de imagem, de forma que cada linha possa ser lida a seu tempo.

A transmissão de dados compreende três tipos: TSL (Transmissão por Salto de Linha), TB (Transmissor de Buffer) e MH (Modified Huffman). Como o tempo de transmissão depende da quantidade de informações na página, o TSL ignora as entrelinhas e emite um comando para saltá-las. O MH permite economizar, comprimindo os dados por meio de códigos de redundâncias, o que também abrevia o tempo de transmissão.

Ainda recém-comercializada, a IFAX 3021 foi adquirida pela Empresa Brasileira de Correios e Telégrafos para atender à rede de agências que operam o post-grama. Em 1985, esse serviço se estendia a 45 cidades em território nacional e a 24 países. Para as cidades brasileiras, o prazo de entrega do documento é de 1 hora e 30 minutos. Para o exterior, chega ao destino em 12 horas, quando expedido antes do meio-dia (horário nacional). A transmissão é feita entre correios, e o custo das tarifas nacionais varia conforme a cidade.

Simplicidade de operação
Da esquerda para a direita, vemos as diversas fases de funcionamento da IFAX 3021: primeiro, os documentos originais são colocados na bandeja voltados para baixo. Depois, chama-se pelo telefone o número da máquina receptora. Basta então acionar a tecla [Início] após o sinal de recepção. Esta é automática e não depende da presença de um operador.



CONTROLE AUDITIVO

Variações

Este programa foi escrito para o TK 90X. As instruções PRINT pressupõem o uso de um vídeo com 32 colunas, devendo ser alteradas se você quiser adaptá-las para um micro com tela de quarenta colunas.
Os comandos SOUND, RAND e RND são específicos do BASIC do TK 90X.

A seguir apresentamos as alterações necessárias para adaptar o programa ao CP 400 Color.

- Altere a sintaxe das instruções PRINT, deixando-as coerentes com o BASIC do CP 400.
- Elimine a linha 20.
- Altere a linha 190 para:
  190 PRINT D: IF D 254 THEN D = 254
- Altere a linha 200 para:
  200 SOUND (255-D),1

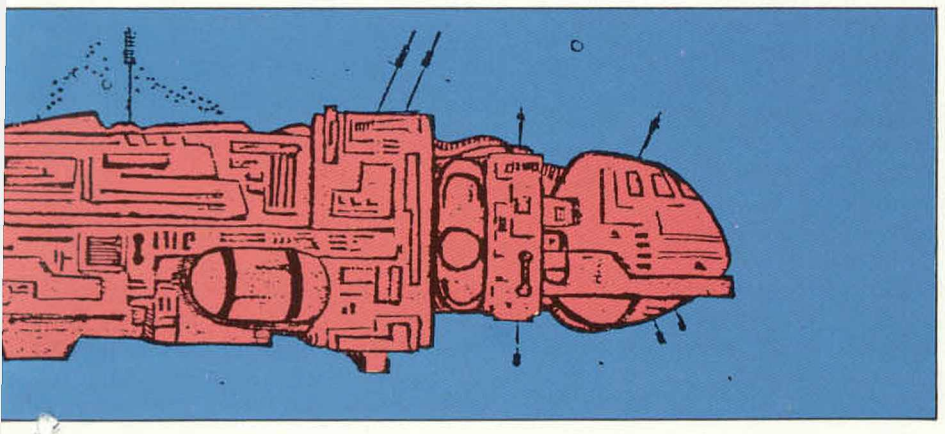
A maioria dos jogos eletrônicos utiliza os efeitos visuais do microcomputador. Apresentamos aqui um que se baseia na capacidade do micro de produzir diferentes sinais sonoros.

Os jogos do tipo fliperama usam sons para complementar seus vibrantes efeitos visuais, aumentando o realismo e a excitação. Raramente a sonorização constitui o tema central de um jogo. Contudo, da mesma forma que fazemos valer nosso sentido de visão, nada impede que usemos a audição como sua base.

Sinal de Alerta utiliza de forma interessante os recursos sonoros. A tarefa, que à primeira vista parece fácil, transforma-se em fascinante desafio, pois exige um sentido pouco usado nos jogos de ação, com seus movimentos rápidos e multicoloridos.

Você comanda uma nave espacial cujo combustível está terminando. Para sobreviver, precisa chegar a uma estação reabastecedora nas proximidades. Mas seu sistema de orientação por computador está com defeito e você não dispõe de indicação visual para se guiar na aterrissagem. Sua única referência é um sinal sonoro, emitido pela estação, que vai se tornando mais agudo à medida que prossegue a aproximação. Portanto, ouça com atenção e reaja rapidamente, acionando as teclas de controle: [I], para cima; [M], para baixo; [K], para a direita; e [J], para a esquerda.

Tal como sucede nas naves verdadeiras, uma vez impulsionada, ela prosseguirá na mesma direção até que você compense o empuxo inicial com um comando oposto. Se utilizar duas vezes [I] para subir rapidamente, terá de teclar [M] duas vezes a fim de parar. Esse fator torna o jogo muito mais difícil do que parece à primeira vista.



Você vai precisar de muita prática para conseguir pousar apenas com os sinais sonoros. Contudo, acrescentando algumas linhas ao programa, será incluída uma indicação visual na tela, que mostrará a posição da nave em relação à estação. Ela pode mostrar a distância que o separa da estação (a variável D) ou indicar que teclas devem ser pressionadas — por exemplo, se SGN (p − x) = − 1, então você deve virar à esquerda. Essas dicas facilitariam bastante o jogo.

Após experimentar nosso programa, talvez você queira criar seus próprios jogos. Usar os sons para localizar objetos ou desviar-se deles é um desafio que oferece muitas possibilidades ao programador. Que tal um jogo para submarinos com sonar, ou um campo minado que você tem de percorrer utilizando um detector que emite alarmas sonoros?

Será vantajoso programar esses jogos para micros com maiores recursos sonoros, como o TK 90X ou o CP 400 Color. Podem-se utilizar vários sons ao mesmo tempo, cada um com seu significado; o volume, programável em algumas máquinas, seria usado como recurso extra.

```
10  REM ** SINAL DE ALERTA **
        *** PARA O TK 90X ***
20  RAND
30  PRINT "MOVA SUA NAVE PARA
    ESTACAO REABASTECEDORA ANTES
    QUE O COMBUSTIVEL TERMINE"
40  PRINT : PRINT "A ESTACAO EMITE UM
    SINAL SONORO. QUANTO MAIS
    PERTO VOCÊ CHEGAR, MAIS AGUDO
    SERA O SINAL": PRINT "I (SOBE)", "M
    (DESCE)","J (ESQUERDA)","K
    (DIREITA)"
50  PRINT: PRINT "PARA INICIAR DIGITE
    UMA TECLA"
70  LET A$ = INKEY$: IF A$ = " " THEN
    GOTO 70
80  LET P = INT (RND * 100) + 1: LET Q =
    INT (RND * 100) + 1: LET F =
    3 * (P + Q)/2: LET X = 0: LET Y = 0
90  LET XD = 0: YD = 0
100 PRINT "COMBUSTIVEL: ";F;TAB (0);"
    DISTANCIA: ";
110 IF (ABS (P − X) < 2 AND ABS (Q − Y) < 2)
    THEN PRINT 0: PRINT "VOCÊ CHEGOU
    A BASE COM";F;"GALOES DE
    COMBUSTIVEL": STOP
120 LET A$ = INKEY$: IF A$ = "" THEN
    GOTO 120
130 IF (AS$ = "I" AND YD < 3) THEN LET
    YD = YD + 1
140 IE (AS$ = "M" AND YD > ı3) THEN LET
    YD = YD − 1
150 IF (A$ = "J" AND YD > − 3) THEN LET
    XD = XD − 1
160 IF (A$ = "K" AND YD < 3) THEN LET
    XD = XD + 1
170 LET X = X + XD: LET Y = Y + YD
180 LET D = SQR ((P − X) * (P − X) +
    (Q − Y) * (Q − Y))
190 PRINT D: IF D > 129 THEN LET
    D = D − 129
200 SOUND ,4,(69 − D)
210 LET F = F − ABS (XD) − ABS (YD): IF
    F > 0 THEN GOTO 90
230 PRINT TAB (0);"
    > < > < > < > < > < > CRASH
    < > < > < > < > < > < >          ": FOR G = 1
    TO 50: PRINT AT 21,0;"
                ";AT 21,0;"   COMBUSTIVEL
    ESGOTADO": NEXT G
240 PRINT : PRINT "VOCE FICOU PERDIDO
    NO ESPACO A      ";D;" UNIDADES
    ASTRONOMICAS", "DA BASE"
260 STOP
```

C

CRIAÇÃO DE REGISTROS

Muitos sistemas gerenciadores de bancos de dados possuem uma linguagem embutida. Procedimentos escritos "sob medida" nessas linguagens aumentam ainda mais a eficiência dos SGBDs.

Um banco de dados faz bem mais do que apenas organizar os registros de um arquivo. É possível ter acesso a registros por meio de comandos simples, mas os melhores sistemas gerenciadores de bancos de dados (SGBDs) permitem ao usuário escrever procedimentos utilizando suas próprias linguagens incorporadas. Isso amplia bastante a eficiência e a utilidade de um banco de dados.

Para examinar o uso em bancos de dados de procedimentos escritos pelo usuário, devemos primeiro rever a maneira pela qual a maioria dos SGBDs utiliza comandos — SEARCH, LOCATE, DISPLAY, NEXT, LAST e assim por diante. Esses comandos são processados pelo SGBD para descobrir o que o usuário deseja. O sistema gerenciador abre o arquivo do banco de dados e o percorre do primeiro ao último registro, selecionando no caminho os registros que satisfaçam os requisitos. Costuma-se dizer que esses comandos pertencem a uma "linguagem de consulta".

Em contraste com os comandos diretos ou de consulta, muitos SGBDs permitem que programas ou procedimentos sejam escritos pelo usuário, simplificando o processo de obtenção e classificação das informações. Esses procedimentos são mais que simples conjuntos de comandos padrão ou de consulta: vêm escritos em linguagens de programação completas, embora limitadas, incluídas no SGBD. Algumas dessas linguagens assemelham-se às mais comuns de programação e não são difíceis de aprender.

O dBase II, da Ashton Tate, por exemplo, incorpora uma linguagem fácil de aprender e de empregar. Suas estruturas mostram algumas semelhanças às do BASIC; razoavelmente simples, incluem DO WHILE-ENDDO, DO CASE-ENDCASE e IF-ENDIF. Além disso, existem cerca de cem comandos — alguns familiares, outros desenvolvidos especificamente para uso em bancos de dados. Entre os comandos familiares estão: CALL (passar o controle para uma rotina em linguagem de máquina); NOTE (equivalente ao REM do BASIC); STORE <expressão> TO <variável> (equivalente a LET <variável> = <expressão> em BASIC); e muitos outros. Comandos menos familiares incluem o SKIP (avanço ou retrocesso pelos registros do arquivo) e o FIND <string de caracteres>.

Também são fornecidas várias funções, algumas semelhantes às do BASIC. Por exemplo: CHR <expressão> (que equivale a CHR$(x)); LEN <expressão em string de caracteres> (equivalente a LEN em BASIC); TYPE <expressão> (retorna o tipo da expressão); e DATE() (uma variável do sistema contendo a data).

Para extrair informações, um procedimento feito sob medida é mais eficaz que uma seqüência de comandos introduzidos a partir do teclado. Podemos comprovar isso criando um banco de dados de estoque e inventário com o dBase II e escrevendo um procedimento nessa linguagem. Os registros no arquivo serão muito simples, com quatro campos para caracteres numéricos e um para não numéricos:

```
CÓDIGO DO FABRICANTE                06116
CÓDIGO DA PEÇA                       3995
QUANTIDADE                             86
PREÇO                              34,750
DESCRIÇÃO              DISCO DE FREIO
```

Em primeiro lugar, precisamos encontrar uma só palavra para cada campo, estabelecer os valores que queremos para eles e determinar se são de caracteres "não numéricos", "numéricos" ou "lógicos". Isto será suficiente:

```
CODFABR     : 5 caracteres; numérico
CODPECA     : 5 caracteres; numérico
QUANTIDADE  : 3 caracteres; numérico
PRECO       : 6 caracteres; numérico
DESCRICAO   : 40 caracteres; não numérico
```

Quando o dBase II começa a rodar, surge um ponto na tela indicando que você deve entrar com os dados. O comando CREATE inicia um novo arquivo, que denominaremos PECAS, por exemplo. O processo será assim:

```
. create PECAS <CR>
ENTRAR COM ESTRUTURA DO REGISTRO
  COMO SEGUE:
CAMPO NOME, TIPO, LARGURA, CASAS
  DECIMAIS
001 codfabr,n,5,0
002 codpeca,n,5,0
003 quantidade,n,3,0
004 preco,n,6,2
005 descricao,c,40
006 <CR>
```

Nossos dados foram introduzidos em minúsculas, e a resposta do dBase II foi impressa em maiúsculas. Repare que o número de casas decimais dos campos numéricos — bem como o ta-

manho do campo — deve ser especificado. Se a tecla [Return] for pressionada sem que se tenha entrado com uma informação — tal como foi feito com o campo 006 —, a fase inicial da criação de arquivos será encerrada.

Utilizamos, a seguir, o comando APPEND. Tem o efeito de limpar a tela e exibir um "esqueleto" do registro, à espera de dados.

```
CODFABR    :      :
CODPECA    :      :
QUANTIDADE :   :
PRECO      :           :
DESCRICAO:
```

Podemos, agora, digitar os dados para cada campo. Um [Return] indica que se completou a entrada num campo; logo que um [Return] for pressionado para o último campo, todos os dados serão aceitos, e o esqueleto do registro reaparecerá, esperando pelo conteúdo do registro seguinte. Para terminar a entrada de dados, aperta-se [Return] no início de um registro vazio.

Introduzidos os dados no arquivo, vamos ver agora como utilizá-lo. Suponhamos que você precise saber em detalhe o valor de varejo de todo o seu estoque, para fins contábeis. Você poderia ver na tela, um a um, todos os registros, e anotar o número de peças ainda em estoque; a seguir, registraria o preço unitário e multiplicaria os dois números. Depois de percorrer todos os registros, o valor total seria obtido pela soma de todos os subtotais. Se você estiver utilizando um fichário convencional, essa poderá ser a única maneira de chegar a uma solução.

A maioria dos SGBDs permite obter esse tipo de informação de modo muito mais simples. Eis como poderia ser feito com o dBase II:

```
.use pecas
.sum quantidade * preco
37.870,58
```

Na primeira linha, um comando instrui o dBase II para trabalhar com o arquivo denominado PECAS. A segunda linha pede para "somar o resultado da multiplicação do campo QUANTIDADE pelo campo PRECO, em cada registro". Na terceira, você vê o tipo de resposta que poderá aparecer após o valor total, em dinheiro, do estoque todo. Melhor que o fichário, não?

Esse exemplo ilustra com clareza como pode ser útil um SGBD eficiente, mesmo utilizando apenas um simples comando. Mas sua capacidade efetiva se manifesta quando armazena seqüências de comandos sob a forma de programas. Para armazenar num programa — que chamaremos de VALOR — nossos comandos simples de avaliação de estoque, seria necessário o seguinte diálogo:

```
.modify command
ENTRE NOME DO ARQUIVO: valor
set talk off
use pecas
sum quantidade * preco
set talk on
cancel
```

Esse curto programa será armazenado em disco e poderá ser usado digitando-se o comando DO:

```
.do valor
```

O programa será lido e executado automaticamente. Tão logo seja encerrado, o controle retornará ao dBase II.

Procedimento de consultoria

```
* Mostra consultores
SET TALK OFF
USE SOCIOS
DO WHILE.NOT.EOF
  IF PROFISSAO = "CONSULTOR"
    DISPLAY NOME
  ENDIF
  SKIP
ENDDO
```

Este programa em dBase II extrai os nomes dos consultores participantes de um clube. A linha inicial, com um asterisco, indica que se trata de um comentário. Não precisamos que todos os números de registro sejam exibidos à medida que se efetua a busca. Por esse motivo, comandamos SET TALK OFF.

Devemos especificar que o arquivo procurado é o SOCIOS, pois podem existir numerosos arquivos diferentes no banco de dados. Para que o dBase II procure através dos registros disponíveis, construímos um loop em torno da éstrutura DO WHILE-ENDDO.

Dentro do loop, uma condição estabelece que, quando o campo profissão contiver o string "CONSULTOR", o programa exibirá o campo do nome do registro. A condição IF deve ser encerrada por ENDIF. O programa é então entregue a um comando SKIP, que diz ao dBase II para ir ao registro seguinte.

O PROTOCOLO MIDI

Excelente auxiliar nas primeiras experiências com música eletrônica, o seqüenciador analógico foi suplantado pelo protocolo MIDI, que deu aos músicos mais versatilidade e recursos quase ilimitados.

O seqüenciador analógico causou enorme impacto na composição de música eletrônica, nas apresentações ao vivo e nas gravações de discos. Mas ele apresentava a desvantagem de controlar uma gama de variáveis em apenas um sintetizador. Um músico que, por exemplo, possuísse dois sintetizadores — um com poderosa capacidade de seqüenciação e outro com boa qualidade de som — não poderia conectá-los de modo a combinar suas características num único equipamento. O mesmo ocorria nos estúdios de gravação, que não dispunham de um meio de coordenar uma grande variedade de sons gerados por diferentes equipamentos.

A tentativa de solução para esse problema levou à criação de uma interface digital para centralizar o controle de todo o sistema numa única máquina. A MIDI (Musical Instrument Digital Interface, "interface digital para instrumentos musicais") tornou possível, na produção musical, o controle de qualquer sistema digital por outros equipamentos, como, por exemplo, microcomputadores.

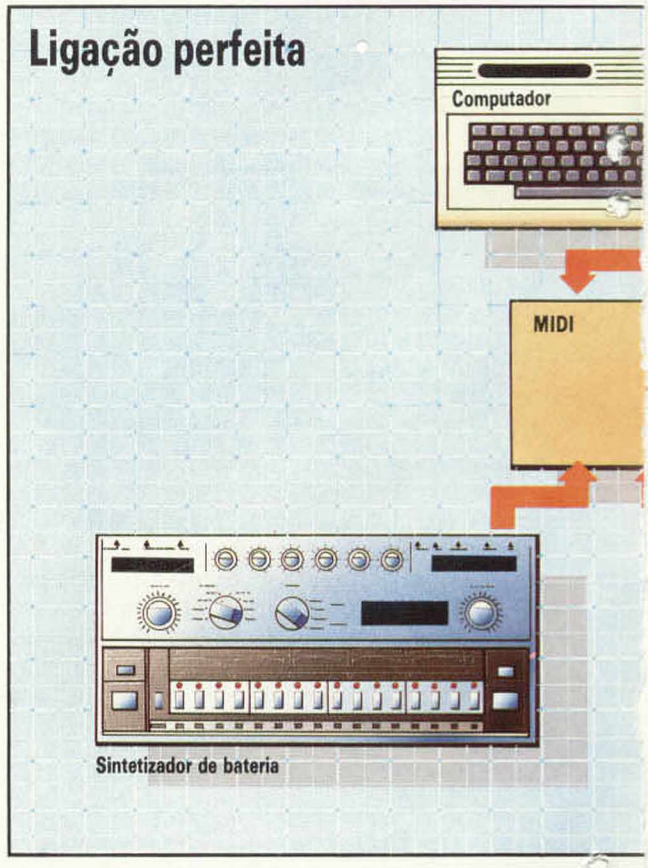
Desenvolvido em função de um acordo estabelecido entre os principais fabricantes de instrumentos digitais do Japão e dos Estados Unidos, o atual protocolo, ou especificação, foi concluído em agosto de 1983. Trata-se de um circuito de controle que transmite dados de um instrumento para outro ou de um microcomputador para um instrumento musical.

A MIDI funciona com um processador de 8 bits, e seus projetistas foram obrigados a escolher entre os dois modos de transmissão disponíveis — paralelo ou serial. Na transmissão paralela, é necessária uma linha para cada bit, de forma que os 8 bits de cada byte são enviados simultaneamente, possibilitando elevada velocidade de transmissão. Suas desvantagens estão no alto custo e na inconveniência de se terem ligações de, no mínimo, oito fios ligados a um conector de 25 pinos.

A transmissão serial, por outro lado, emprega apenas duas linhas: numa delas, enviam-se os bits de dados um atrás do outro; a outra serve para que o instrumento receptor acuse erros de paridade no fluxo de dados. Embora mais lenta que a paralela, a transmissão serial utiliza conector mais simples e é bem mais barata.

A motivação inicial para a criação da MIDI foi a necessidade de um circuito de controle que realçasse a música sintetizada e proporcionasse algum tipo de comando de tonalidade. Além disso, o preço deveria ser acessível aos proprietários de microcomputadores e músicos que usam sintetizadores e baterias eletrônicas. Por estas razões, os fabricantes decidiram pelo uso da transmissão serial na MIDI.

Essa interface transmite em modo assíncrono, como a RS232C (padrão de muitos microcomputadores para conexão a modems ou impressoras seriais). A transmissão realiza-se por intermédio de um chip Motorola 6850 ACIA (Asynchronous Communications Interface Adaptor, "adaptador para interface de comunicações assíncronas"), que acrescenta 2 bits a cada byte do instrumento dominante. O primeiro é "0" (bit de partida), seguido pelos 8 bits do dado serial, terminando tudo com "1", o bit de parada. Essa palavra serial de 10 bits é então enviada ao instrumento receptor, onde um segundo chip 6850 a converte aos 8 bits originais.



Glissando
Para conseguir o glissando num instrumento de cordas, basta deslizar os dedos sobre elas até a nota seguinte. Mas, para produzir o mesmo efeito sonoro num sintetizador, é preciso uma programação bastante complexa.

Por ser caro, esse chip fica protegido no circuito por opto-isoladores (dispositivos dotados de células fotoelétricas que permitem a troca de sinais entre dois circuitos eletricamente isolados); assim, os picos de tensão não provocam qualquer dano ao microprocessador.

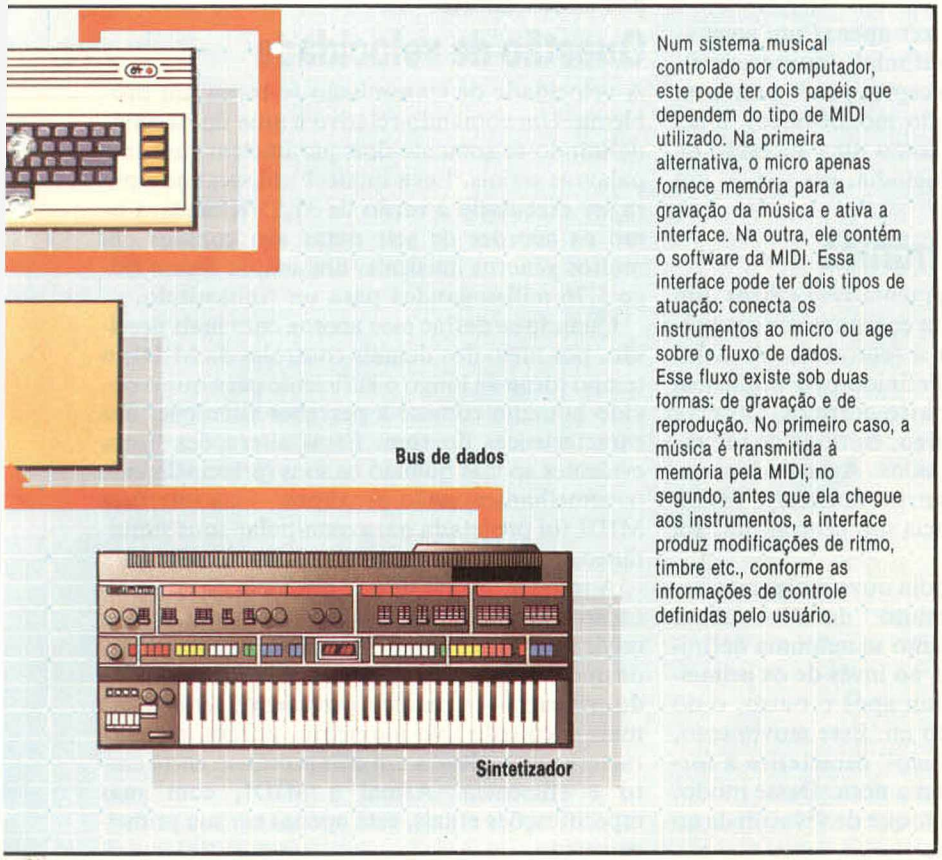
A MIDI difere da interface RS232 num ponto importante: a velocidade de transmissão da RS232 chega a 19,2 Kbauds, enquanto a da interface para instrumentos musicais é somente a metade. Isso não afeta a compatibilidade com os microcomputadores, pois os circuitos lógicos internos da MIDI ajustam a velocidade de sincronização para 31,25 Kbauds (relativamente alta para transmissões seriais).

Como a interface MIDI coordena mais de um instrumento, sua primeira tarefa é enviar o dado certo ao instrumento apropriado — caso contrário, uma bateria eletrônica acabará tocando uma melodia cuidadosamente seqüenciada, enquanto um sintetizador polifônico tentará reproduzir uma batida sincopada de bateria.

Em função disso, os instrumentos compatíveis com a MIDI dispõem de um código de identificação numérico. Um dos dezesseis canais disponíveis na interface destina-se ao código, de forma que somente esse canal aceita dados para determinado instrumento. Assim, a primeira parte de uma transmissão completa é um byte de status que inclui o identificador. Todos os dados que seguem essa instrução de encaminhamento especificam, portanto, como o comando deve ser interpretado pelo instrumento.

A interface possui dois conectores DIN de cinco pinos, denominados MIDI IN e MIDI OUT. O primeiro recebe instruções de um microcomputador ou sintetizador mestre e o outro transmite a seqüência modificada de bits para o instrumento receptor.

Muitos modelos contam ainda com o conector MIDI THRU, uma saída que transmite sem alterações o fluxo de bits (enviado para o MIDI IN). Esse fluxo pode então ser dirigido para uma segunda interface. O cabo, com 15 cm de comprimento máximo e conectores DIN de cinco pinos, possibilita a ligação a um microcomputador ou ao sintetizador mestre.



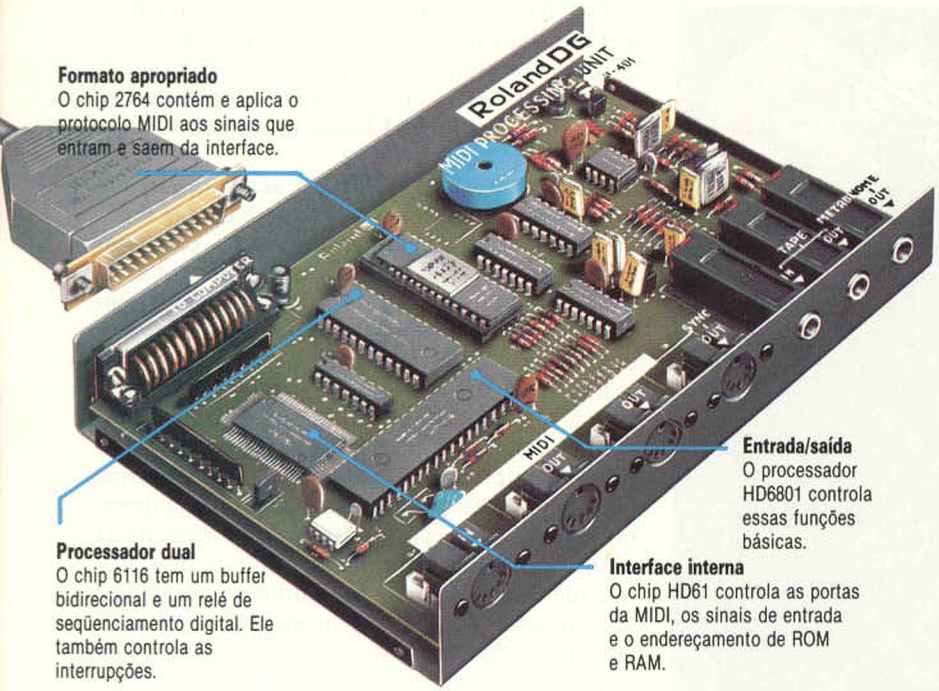
Num sistema musical controlado por computador, este pode ter dois papéis que dependem do tipo de MIDI utilizado. Na primeira alternativa, o micro apenas fornece memória para a gravação da música e ativa a interface. Na outra, ele contém o software da MIDI. Essa interface pode ter dois tipos de atuação: conecta os instrumentos ao micro ou age sobre o fluxo de dados. Esse fluxo existe sob duas formas: de gravação e de reprodução. No primeiro caso, a música é transmitida à memória pela MIDI; no segundo, antes que ela chegue aos instrumentos, a interface produz modificações de ritmo, timbre etc., conforme as informações de controle definidas pelo usuário.

O dó central

Imagine alguém que usa pela primeira vez a MIDI e pretende tocar uma melodia curta. Esta começa com um dó central, sobe para mi, depois para sol e assim por diante. O modo de programar depende do tipo de software musical em uso. As notas são introduzidas com uma caneta óptica na pauta de cinco linhas apresentada no vídeo. Ou, então, por meio do teclado do micro, segundo algum tipo de MCL (Musical Composition Language, "linguagem de composição musical"). Uma alternativa seria tocar as notas num teclado musical, ligado ao micro, que, sem gerar som algum, produzisse alguma das representações resumidas acima.

Introduzida a música, a transmissão da MIDI será sempre a mesma: para iniciar a melodia, o primeiro byte — transmitido como mensagem serial com seus 2 bits extras a partir do chip ACIA — seleciona, por exemplo, o canal 6; o segundo byte, o dó central.

Essa instrução produzirá a nota dó central no sintetizador, que a reproduzirá sem parar, a menos que haja outra instrução para interromper ou limitar sua duração. Não havendo, e se o resto



Formato apropriado
O chip 2764 contém e aplica o protocolo MIDI aos sinais que entram e saem da interface.

Entrada/saída
O processador HD6801 controla essas funções básicas.

Processador dual
O chip 6116 tem um buffer bidirecional e um relé de seqüenciamento digital. Ele também controla as interrupções.

Interface interna
O chip HD61 controla as portas da MIDI, os sinais de entrada e o endereçamento de ROM e RAM.

MIDI
Essa interface identifica e coordena os protocolos de entrada/saída de um computador e dos instrumentos musicais ligados a ele. Também processa o som digitalizado, incluindo, nos dados que o definem, informações de controle, sincronização e ritmo para serem transmitidas a um sintetizador.

da melodia for introduzido sem parâmetros de duração, todas as notas continuarão a soar ao mesmo tempo. O resultado será um acorde ininterrupto, composto de todas as notas da melodia.

Um mínimo de familiaridade com o pentagrama na tela basta para detectar os eventuais enganos que forem cometidos. Assim, para corrigi-los, um usuário de MCL executaria a seqüência desejada enviando uma instrução monofônica ao sintetizador.

"Monofonia" quer dizer apenas um som — em contraposição a "polifonia" (muitos sons). Quando o sintetizador receptor produz um som por vez (opção pelo modo monofônico), a seqüência será executada como uma sucessão de notas isoladas — uma melodia, em vez de um acorde.

Composição eletrônica

Suponha que, após várias tentativas e erros, um principiante tenha digitado corretamente sua música e que o sintetizador a reproduza. A melodia mantém o ritmo — definido pela seqüência de durações — e o compasso corretos. Observe que, até agora, neste artigo, os tipos de instrução têm sido muito limitados. Apenas duas características (ou parâmetros musicais) foram mencionadas — freqüência (dó central, mi, sol etc.) e duração.

O compositor da melodia ouve-a algumas vezes e acha que ela soa muito "dura" — como é provável ocorrer enquanto se tem uma definição mínima. Decide que, ao invés de os primeiros dó e mi ocorrerem um após o outro, o dó deve durar até o início do mi. Esse movimento, conhecido como "glissando" caracteriza a forma pela qual se alonga uma nota. Desse modo, o usuário acrescenta um toque de vivacidade ao trabalho do sintetizador.

Isso conduz a um novo problema: se o sintetizador receptor não possui recursos para realizar glissandos, não executará a instrução. Poderá tocar o dó central como se estivesse recebendo a instrução original, ou fazer algo totalmente diferente.

Se o usuário da MIDI der instruções para a produção de um trecho de música polifônica e o sintetizador for monofônico, este fará uma imprevisível seleção da polifonia e a executará monofonicamente. Em resumo, o uso da MIDI para acoplar um micro a um sintetizador extremamente elementar não irá transformá-lo num equipamento sofisticado como o Fairlight.

Essas restrições valem também para o processo inverso: o sintetizador receptor pode ter excelente qualidade, mas, a não ser que se definam parâmetros musicais suficientes — e a menos que se ajustem corretamente os próprios controles do sintetizador —, o resultado será equivalente à musicalidade de uma calculadora de bolso.

Na prática, é possível amenizar essa incompatibilidade: por meio dos controles do sintetizador, estabelecemos como constantes o maior número possível de parâmetros, para que as instruções da MIDI se mantenham de acordo com eles. Essa abordagem tem maior probabilidade de ser adotada pelo usuário do sintetizador.

Para enriquecer a execução, a interface tem, em tese, 128 controles — filtragens, distorção, "ruído branco" (todas as freqüências possíveis) e "ruído rosa" (freqüências médias), cada um com valores entre 0 e 128 —, suficientes para abranger os parâmetros disponíveis na maioria dos sintetizadores.

Questão de velocidade

A velocidade de transmissão se torna um problema. Um comando relativo a uma única nota, definindo-se somente dois parâmetros, usa três palavras seriais. Leva quase 1 milissegundo para ser executado à razão de 31,25 Kbauds. Como os acordes de seis notas são comuns em muitos gêneros musicais, um acorde desses exige 5,76 milissegundos para ser transmitido.

Quando se define esse acorde com mais precisão, por meio dos demais controles da MIDI, o tempo torna-se longo o suficiente para que o ouvido humano comece a perceber alterações nas características do som. Essas alterações ficam evidentes apenas quando os sons (principalmente os semelhantes) estão próximos — e a interface MIDI foi projetada para manipular sons simultâneos.

A música, no entanto, é um meio de comunicação "paralelo": as pessoas acostumaram-se a ouvir sons simultâneos. Portanto, não surpreende que a transmissão serial da interface tenha sido criticada — a paralela certamente seria muito mais adequada. No momento, porém, essa característica se deve a um compromisso entre custo e eficiência. Afinal a MIDI, com suas especificações atuais, está apenas em sua primeira versão.

OSBORNE ENCORE

Fabricante do primeiro micro portátil, a Osborne recuperou-se de seus problemas financeiros e lançou o Encore — mais um integrante da extensa linha de máquinas compatíveis com o IBM PC.

Primeiro computador portátil integrado, o Osborne-1 iniciou uma revolução na microcomputação. Equipado com monitor incorporado, duas unidades de disco e interfaces para modems e impressoras, foi também o primeiro computador profissional autônomo operando com CP/M. Embora sua classificação como "portátil" fosse um tanto exagerada (o Osborne-1 pesava 10,5 kg), outros fabricantes logo perceberam o potencial da nova máquina. Em pouco tempo, a Osborne estava rodeada de concorrentes.

O aviso de novos tempos veio em 1981, quando a IBM anunciou o lançamento de seu primeiro modelo na linha de computadores pessoais. A entrada da gigante americana na produção de micros arrebatou enorme fatia do mercado, atraída pela fama da empresa. Se, por um lado, os micros da IBM provocaram o desaparecimento de algumas marcas, por outro, representaram enorme avanço na indústria de computadores, estimulando outros fabricantes a produzirem equipamentos compatíveis com o IBM. Tal foi o caso da Osborne, que, juntamente com outros competidores, não perdeu tempo e anunciou um novo modelo, o Osborne Executive, compatível com o IBM PC.

Esse equipamento vinha com dois processadores, permitindo executar programas aplicativos para CP/M e para MS-DOS. No entanto, devido à escassez mundial do microprocessador 8086, a máquina foi lançada sem o chip necessário para torná-la compatível com o IBM PC. Como resultado, as vendas caíram vertiginosamente, quase obrigando a empresa a fechar suas portas em meados de 1983. Mas a Osborne conseguiu sobreviver, dedicando-se à pesquisa e ao desenvolvimento.

Para uma empresa que quase chegou a falir e mal conseguiu se manter à tona no competitivo mercado de computadores, seu novo lançamento — o Osborne Encore — revela-se uma audaciosa jogada. Mesmo não sendo tão compacto quanto outras máquinas como, por exemplo, o Epson PX-8, o Encore destaca-se por situar-se entre os dois segmentos do mercado: os equipamentos de mesa e os pequenos portáteis.

O Encore pesa 6 kg, bem menos que seu predecessor. O teclado se dobra contra a tela, formando uma caixa compacta, do tamanho aproximado de três listas telefônicas.

O teclado é do tipo máquina de escrever; acima dele há um painel tipo membrana, sensível ao toque, com as teclas de função e os "ícones" (símbolos que representam os programas residentes). O corpo do computador abriga também um visor de cristal líquido de 23,7 x 8 cm. As teclas de controle encontram-se nos dois lados, enquanto as quatro teclas do cursor dispõem-se no canto inferior direito.

Tem-se a impressão de que as teclas se comprimem numa área muito estreita. Isso resulta da tentativa de colocar, num espaço reduzido, todas as características do IBM PC, que tem, à direita, um teclado numérico separado. Esse conjunto, com vistas a manter a compatibilidade, foi incorporado à parte principal do teclado do Encore. As teclas numéricas estão marcadas em azul, contrastando com o branco das alfanuméricas. Para utilizar as numéricas e os outros programas residentes, tocam-se antes os ícones correspondentes. Estes, no Encore, representam as rotinas de cálculo, modem, disco e o programa calendário.

O projeto do painel de toque não foi muito feliz, depreciando o ar profissional do teclado.

Compacto
O Encore da Osborne é uma das primeiras máquinas compatíveis com o IBM PC equipadas com visor de cristal líquido em substituição ao monitor de vídeo convencional. Quando não está em uso, o teclado se fecha contra a tela, formando um volume bastante compacto. Pode-se carregá-lo a tiracolo pela alça.

As teclas de função (que no IBM PC estão separadas à esquerda) são do tipo membrana, desagradáveis ao tato. Embora se possa sentir o movimento de membrana sob o painel, as teclas carecem do toque firme de um teclado profissional.

Incorporando um visor de cristal líquido (LCD) os projetistas obtiveram grande economia no consumo de energia, visto que gasta bem menos que o tubo de raios catódicos usual. Aí residem os maiores problemas de compatibilidade com o IBM PC, não por deficiência nas cores resultantes do visor de cristal líquido, pois estas podem ser facilmente substituídas variando-se o sombreado dos gráficos; a desvantagem está no pequeno tamanho da tela.

A tela normal do IBM PC tem, em modo texto, uma resolução de 80 colunas x 25 linhas; mas, devido aos problemas encontrados pelos fabricantes japoneses no desenvolvimento de um LCD equivalente, a Osborne viu-se forçada a equipar o Encore com um visor de 80 colunas x 16 linhas. Em conseqüência, muitos programas aplicativos escritos para o IBM PC não podem ser utilizados no seu equipamento.

Isso não constitui problema para os programas que rolam verticalmente na tela; mas, em pacotes onde o visor é "paginado", o usuário, em geral, encontrará sérias dificuldades, pois as mensagens aparecem na parte inferior do vídeo. Comparado ao monitor de vídeo, o LCD ainda apresenta outras desvantagens, como a necessidade de forte iluminação para produzir caracteres legíveis e o tempo de familiarização que exige do usuário.

Do lado direito do computador, há espaço para duas unidades de disco flexível de 5 1/4 polegadas; contudo, o modelo básico vem equipado com apenas uma unidade. Do lado oposto situam-se o encaixe para o transformador de energia, uma chave liga/desliga, um botão para ajuste do contraste da tela, e o alojamento das baterias — fontes de energia especiais, de cádmio-níquel, que permitem o funcionamento da máquina por um período de 4 a 5 horas.

Conexões

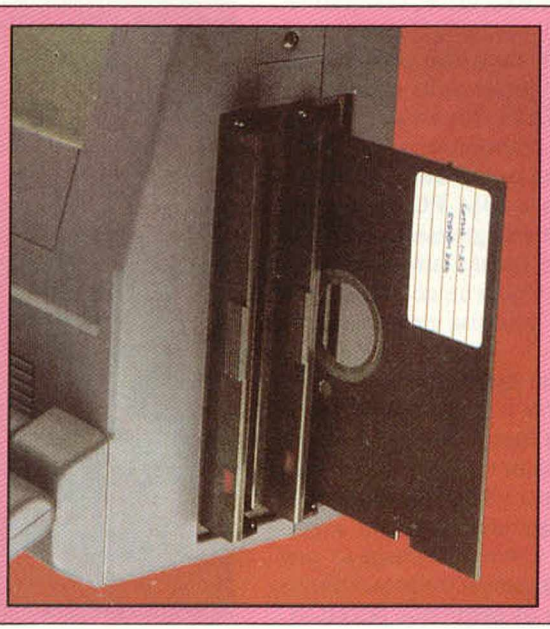
Na parte posterior do Encore estão os conectores: um telefônico, que se liga ao modem interno, uma saída paralela padrão Centronics para impressora e uma saída serial padrão RS232 para conexão com outros dispositivos, como impressoras seriais e modems externos. Para carregar o disco de sistema operacional MS-DOS, basta pressionar no painel de toque o ícone que representa um disco.

Todos os programas em disco do IBM PC podem ser lidos pelo Encore. No entanto, devido às restrições da tela, é difícil detectar se a execução está correta. Entre os programas que o Encore executa de forma bem-sucedida está o Lotus 1-2-3 — difícil de se rodar num compatível, por sua maneira pouco ortodoxa de acessar as rotinas residentes no IBM.

Alto-falante
Recursos visuais são enfatizados por sinais sonoros.

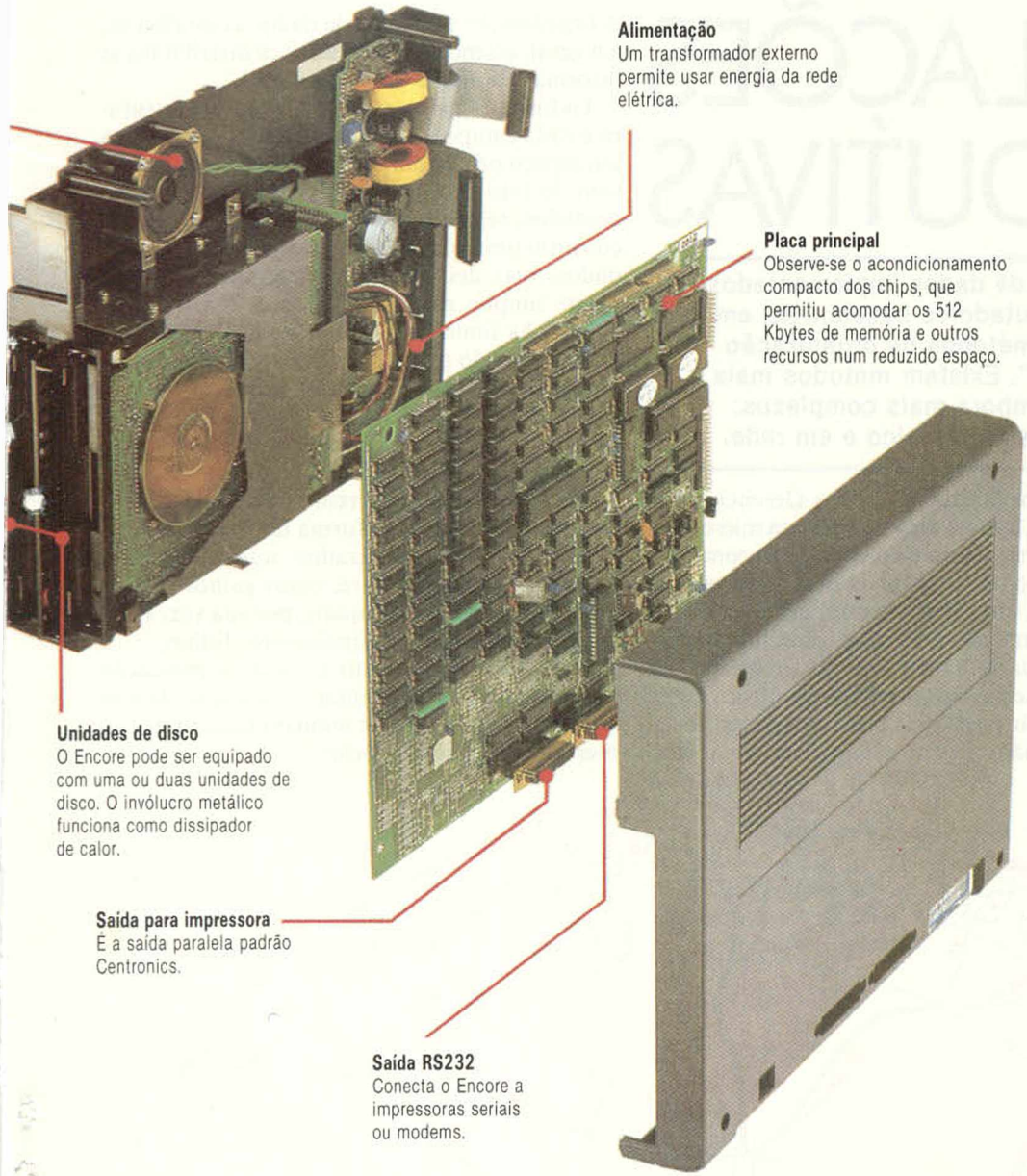
Visor de cristal líquido
O LCD exige muito menos energia que os monitores de vídeo convencionais, tornando realidade o compatível IBM PC acionado a bateria.

Funções duplas
O teclado do Encore contém poucas teclas a menos que o do IBM PC. Mas fornece as mesmas funções agrupando duas delas na mesma tecla.



Compacto, mas incômodo

As unidades de disco (podem ser usadas até duas) situam-se na lateral da máquina, tornando-a muito compacta. Para o usuário, no entanto, isso constitui um aborrecimento, pois obriga-o a esticar-se para inserir um disco ou verificar se a unidade certa está sendo acessada.



Alimentação
Um transformador externo permite usar energia da rede elétrica.

Placa principal
Observe-se o acondicionamento compacto dos chips, que permitiu acomodar os 512 Kbytes de memória e outros recursos num reduzido espaço.

Unidades de disco
O Encore pode ser equipado com uma ou duas unidades de disco. O invólucro metálico funciona como dissipador de calor.

Saída para impressora
É a saída paralela padrão Centronics.

Saída RS232
Conecta o Encore a impressoras seriais ou modems.

OSBORNE ENCORE

MICROPROCESSADOR

8086, de 16 bits.

MEMÓRIA

RAM de 521 Kbytes.

SISTEMA OPERACIONAL

MS-DOS.

VISOR

Visor de cristal líquido. Modo texto: 80 x 16 caracteres; modo gráfico: 480 x 128 pixels.

TECLADO

Com 62 teclas tipo máquina de escrever, mais um painel de toque tipo membrana com dez teclas de função e quatro de ícones.

LINGUAGENS

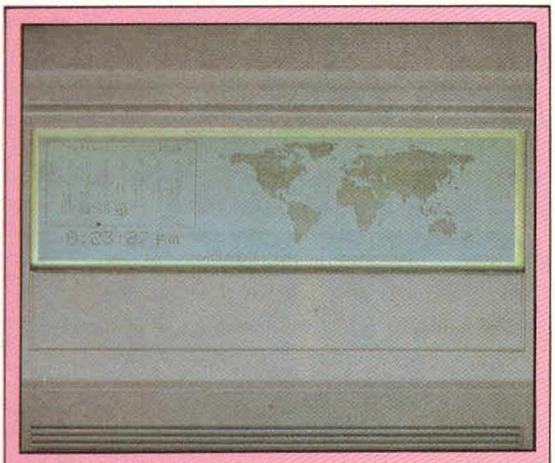
BASIC da Microsoft, fornecido em disco.

INTERFACES

Saída serial RS232, saída paralela padrão Centronics e conector telefônico RJ11.

DOCUMENTAÇÃO

Os dois guias do usuário são, como a maioria dos manuais da Osborne, muito bem escritos, com amplas explicações sobre o funcionamento da máquina e vários exemplos.



Controlador de tempo
Na foto, o visor de cristal líquido de 80 x 16. O espaço delimitado na parte inferior destina-se a acomodar a futura tela de 80 x 25. O visor com calendário/agenda permite registrar compromissos futuros nos diferentes horários das regiões do mapa.

Sensível ao toque
Acima do teclado principal está o painel de toque, tipo membrana — semelhante ao da linha Sinclair —, contendo dez teclas de funções programáveis. Abaixo destas estão os ícones, usados para acionar os programas gravados na ROM do Encore.

RELAÇÕES PRODUTIVAS

Os bancos de dados implementados em microcomputadores baseiam-se, em geral, nos métodos de organização "relacional". Existem métodos mais eficazes, embora mais complexos: os sistemas hierárquico e em rede.

A maioria dos SGBDs (Sistemas Gerenciadores de Bancos de Dados) disponíveis para microcomputadores pertence ao tipo conhecido como sistema "relacional". Em nível bem simples, isso significa que cada registro num arquivo SGBD assume a forma de uma tabela, com linhas e colunas, semelhante a uma planilha financeira. As diversas maneiras de apresentar as informações podem sugerir estruturas mais complexas, mas, em última análise, estas não passam de tabelas.

A organização do banco de dados assemelha-se, em geral, ao modo pelo qual você distribuiria as informações numa folha de papel.

Todos os registros possuem a mesma estrutura e cada campo no interior dos registros ocupa um espaço prefixado. O termo "relacional" provém do fato de estarem as "linhas", no banco de dados, relacionadas às "colunas". Essa não constitui uma forma muito flexível de organizar dados, mas dela resultam programas relativamente simples para manipulação de bancos de dados. As limitações de um sistema relacional fazem parte do preço a ser pago quando se combinam SGBDs e recursos não muito amplos de processamento e capacidade de memória, como os oferecidos pela maioria dos micros — mesmo as novas máquinas de 16 bits.

Uma abordagem diferente consistiria em ordenar os elementos em forma hierárquica. Imagine os dados organizados numa estrutura semelhante a uma árvore, cujos galhos dessem origem a subgalhos, os quais, por sua vez, se dividiriam em ramos e, finalmente, folhas.

Para ilustrar esses dois tipos de organização de dados, vamos apresentar — primeiro de modo relacional, depois de maneira hierárquica — o estoque de um empório:

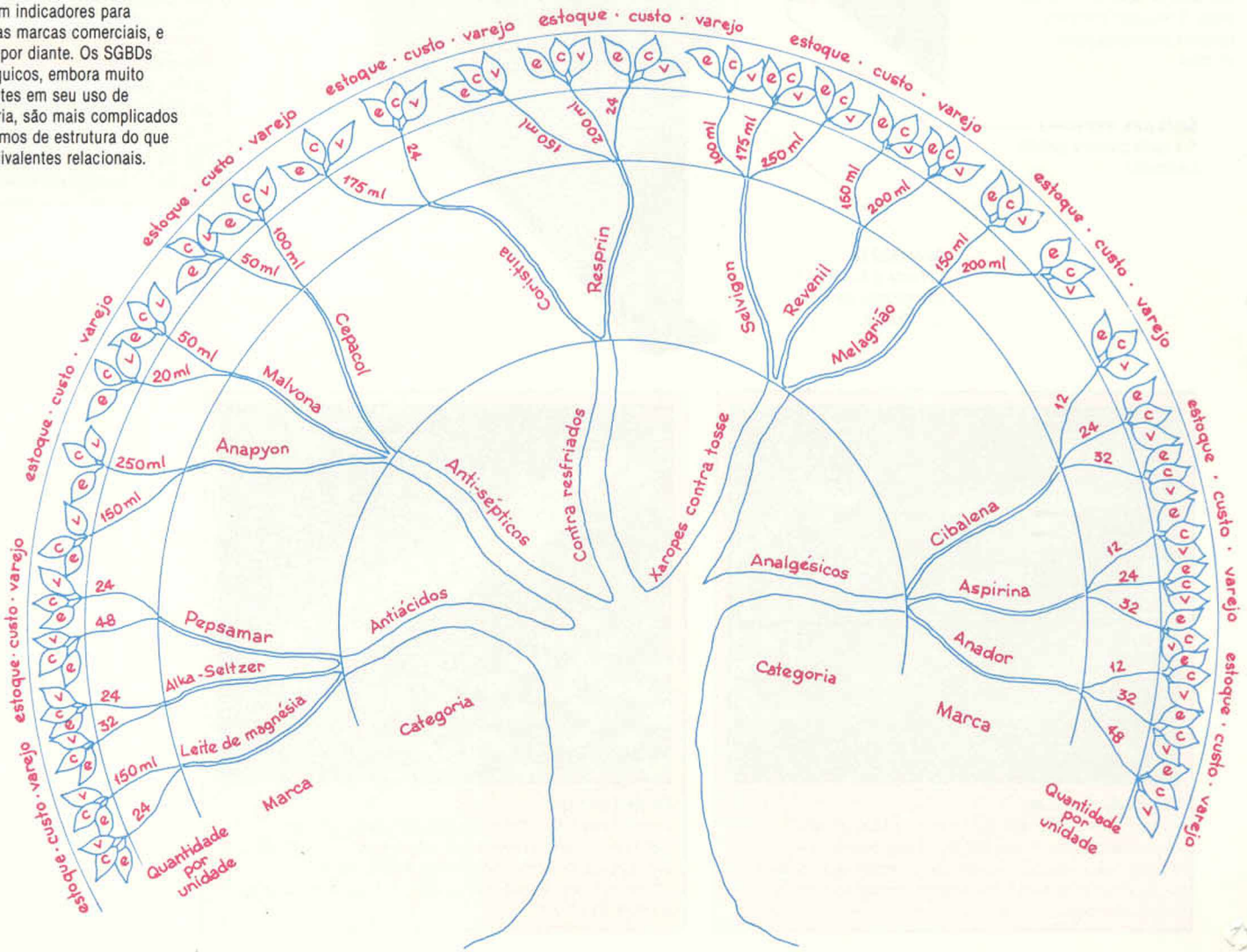
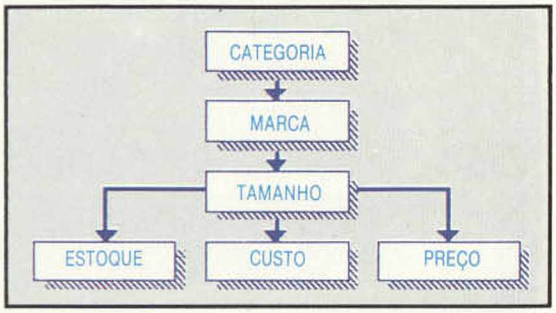


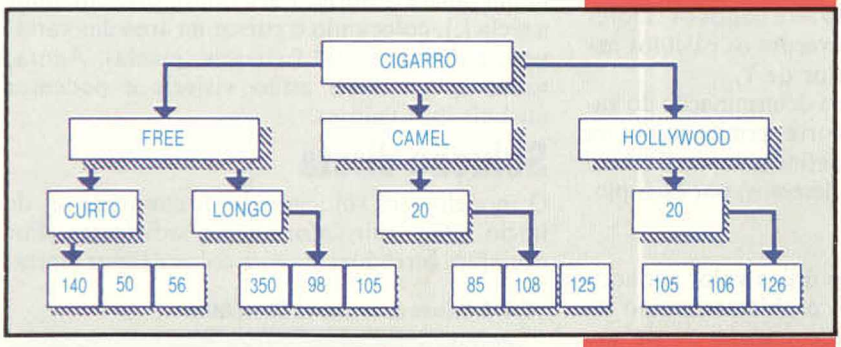
Gráfico em árvore
Enquanto um SGBD "relacional" armazena dados sob forma tabular, um "hierárquico" organiza as informações em forma ramificada. Na figura, a categoria "remédio" contém indicadores para os vários tipos de remédio disponíveis, cada um com indicadores para diversas marcas comerciais, e assim por diante. Os SGBDs hierárquicos, embora muito eficientes em seu uso de memória, são mais complicados em termos de estrutura do que os equivalentes relacionais.

CATEGORIA	MARCA	QUANTIDADE/ UNIDADE	ESTOQUE	CUSTO	VAREJO
CIGARRO	GALAXY	20	350	98	105
CIGARRO	FREE	20	140	50	56
CIGARRO	CAMEL	20	85	108	125
CIGARRO	HOLLYWOOD	20	105	106	126
CHOCOLATE	LACTA	1	106	14	17
CHOCOLATE	GALAK	1	95	12	16
CHOCOLATE	PRESTÍGIO	1	25	15	19
REVISTA	CRIATIVA	1	35	85	95
REVISTA	ELE&ELA	1	12	60	80
REVISTA	MODA	1	86	50	60
BEBIDA	OLD EIGHT	1.000 ml	35	25	37
BEBIDA	SMIRNOFF	500 ml	40	18	27
BEBIDA	COCA-COLA	180 ml	20	16	25

Na representação hierárquica, seria assim o banco de dados para o estoque de produtos:



Ao invés de ser organizado em campos, o registro divide-se em "segmentos" de dados, cada um dos quais contendo mais de um campo. Os itens introduzidos que correspondem a CIGARRO no banco de dados do tipo relacional apresentariam a seguinte organização hierárquica:

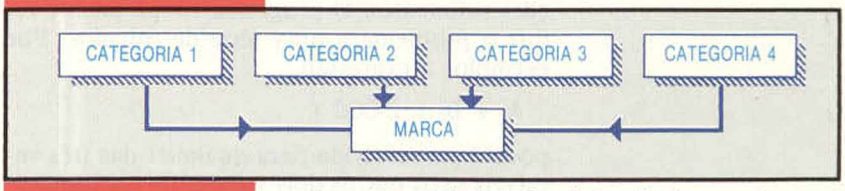


Nesse tipo de representação, é possível reconhecer cada item de informação como superior ou subordinado a outro item. Para isso, precisa-se apenas de indicadores que "apontem" de um segmento de dados para outro. Neste exemplo, precisamos apenas de um segmento CIGARRO, com indicadores para os vários tipos de cigarro em estoque, e indicadores adicionais para o número de cigarros por maço (quantidade por unidade), custos, preços e assim por diante.

Vamos supor que o empório tenha apenas cem "categorias" para a classificação das mercadorias, mas um catálogo de 1.000 artigos à venda; com um banco de dados relacional, seriam necessários 1.000 registros — um para cada artigo. Com um banco de dados hierárquico, porém, apenas cem segmentos abrangem todo o estoque. Isso evita um grande número de duplicações, mas torna o banco de dados estruturalmente muito mais complicado.

Existe outro tipo de organização de banco de dados, um sistema em rede que foi padronizado na CODASYL (Conference on Data Systems Languages, "Conferência sobre Linguagens de Sistemas de Dados"), ficando conhecido com esse nome. O problema dos bancos de dados hierárquicos é que, neles, os dados fluem num único "sentido" (um ramo não pode estar ligado a dois troncos). Na maioria dos casos, isso não acarreta confusões, embora possam existir cigarros com nomes idênticos aos de fábricas de discos etc. As dificuldades aparecem quando uma peça tem mais de um fabricante. Se você vende peças de automóveis, um dispositivo elétrico para um Volks pode ser fabricado pela Peçasmil, Vidrofumê, ou pela Volksmar.

No sistema CODASYL, utiliza-se uma rede de conjuntos, cada um consistindo numa coleção de registros. Ao contrário do sistema relacional, o tamanho do registro não precisa ser fixo. Um conjunto pode incluir apenas um registro, e qualquer registro pode pertencer a mais de um conjunto — desde que não pertença a mais de uma ocorrência do mesmo tipo de conjunto. Assim, não são permitidas estruturas como esta:



Essa é uma limitação séria. Embora sejam mínimas as chances de haver CIGARROS, REVISTAS e BEBIDAS com nomes idênticos, é fácil imaginar uma companhia chamada Supertrigo Ltda., que faça parte de uma estrutura semelhante a esta:

Embora mais flexíveis que os relacionais, os bancos de dados hierárquicos e os do tipo CODASYL são bem mais complexos e requerem, pelo menos, minicomputadores. Os micros utilizam, quase sempre, sistemas relacionais. A dificuldade mais comum decorrente dessa limitação surge quando, por exemplo, se tem um inventário de peças e cada uma tem mais de um fornecedor. Nesse caso, são necessários registros para as peças e um conjunto de registros de fornecedores. Já os SGBDs de multiarquivos permitem que dois ou mais arquivos sejam abertos simultaneamente, e que um arquivo ativo faça referência a um subsidiário.

TENTATIVA E ERRO

Com uma inspeção mais detalhada de alguns de seus recursos exclusivos, conclui-se aqui o exame do programa de processamento de equações TK!Solver, um sucesso no mercado internacional.

O TK!Solver é um software de novíssima geração, que leva o conceito da planilha eletrônica aos domínios da matemática superior e da engenharia. Permite que o usuário defina variáveis com nomes e use esses nomes em equações matemáticas complexas.

Uma das características marcantes do TK!Solver é sua extraordinária capacidade de iterar, ou seja, de chegar a um resultado por ciclos de repetições.

A iteração é um método no qual o programa determina uma variável por meio de tentativas. Em qualquer situação de cálculo, a pessoa que lida com equações poderá determinar os valores de todas as variáveis se tiver, de início, informações suficientes. O programa simplesmente reduz o problema a uma série de cálculos. Por exemplo, a expressão

$$A^2 + B^2 = 2 \cos Y$$

poderá ser resolvida para qualquer das três variáveis se os outros dois valores forem conhecidos. Em face dessa equação — e dados os valores de A e B —, o TK!Solver executa os cálculos necessários e encontra o valor de Y.

Mas há ocasiões em que a determinação do valor não é imediata. Isso ocorre, por exemplo, na equação redundante, que define uma variável em termos dela mesma. Consideremos, por exemplo,

$$D = (A + B)/(2 * D)$$

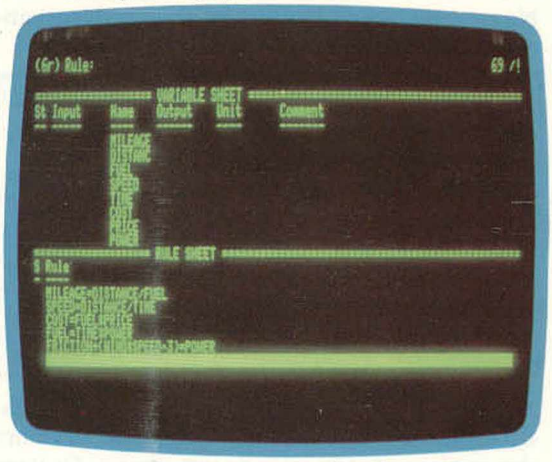
num modelo em que A é o único valor conhecido. Podem ocorrer outros problemas, como resultado de um modelo incompleto ou de um modelo com muitas variáveis interdependentes e uma quantidade limitada de dados.

O modelo da viagem de automóvel, citado em matéria anterior desta série, envolve cinco valores: distância, tempo, velocidade, combustível e autonomia (milhagem). O programa calculava esta última, dados a velocidade e o consumo de combustível; a distância, a partir da velocidade e do tempo; e outras variações simples. Como proceder se quisermos calcular a velocidade que deve ser mantida a fim de completar a viagem dentro de um orçamento predeterminado?

Para começar, acrescentam-se alguns fatores ao modelo. Por exemplo, o modelo tem de considerar a potência do veículo, o atrito interno do motor e a resistência do ar. (Suporemos o atrito interno constante.) Deve-se também estabelecer um teto para o orçamento e o custo do combustível a consumir.

Começaremos a montar o modelo introduzindo essas equações na planilha Regras, uma por linha. As equações são automaticamente transferidas para a planilha Variáveis *(tela 1)*.

Tela 1: Montagem de equações

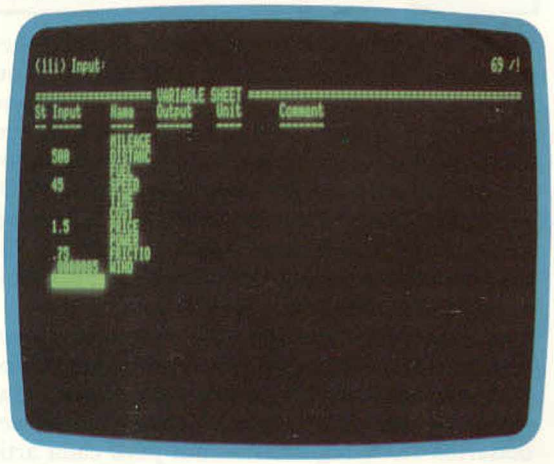


O tamanho da tela é insuficiente para mostrar as muitas variáveis. Podem-se apresentar todas as informações, inserindo a planilha Variáveis numa janela própria. Para tanto, pressionamos a tecla [;], colocando o cursor na área das variáveis, e digitamos W1 (window, janela). Agora, todas as variáveis estão visíveis e podemos atribuir-lhes valores.

Solução direta

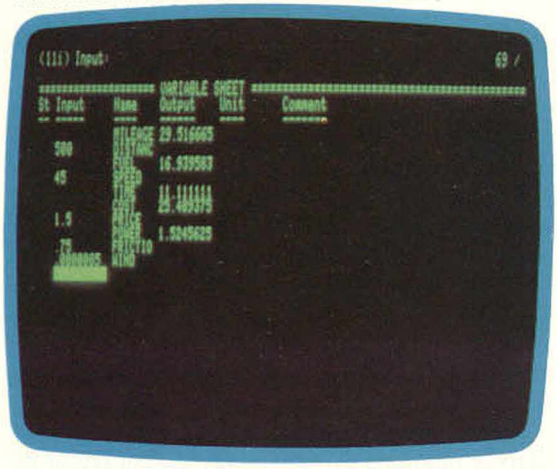
O modelo será solucionado diretamente se, de início, conseguir informações suficientes. Por exemplo, introduzem-se na coluna Input (entra-

Tela 2: Valores de entrada no Direct Solver



da) valores relacionados aos nomes distância, velocidade, preço, atrito e vento (resistência do ar) *(tela 2)*. Pressiona-se, então, [!] para efetuar os

Tela 3: Resultados do Direct Solver

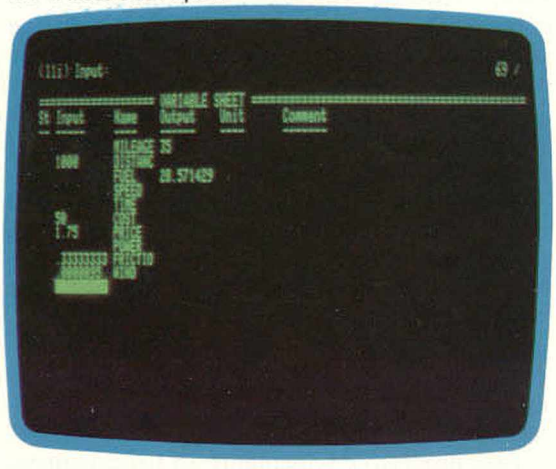


cálculos. O TK!Solver apresenta a mensagem Direct Solver, e os valores das incógnitas surgirão na coluna Output (saída), como se vê na *tela 3*.

Isso dá ao usuário um bom detalhamento das informações desejadas. Mas o que ele fará se quiser usar esse modelo para resolver um problema com menos informações?

Consideremos um cálculo que envolva um máximo de 50 libras para gastar em combustível numa viagem de 1.000 milhas. Sabe-se que o galão de combustível custa 1,75 libra. Assim, determinamos quanto se pode gastar por milha. Começamos apagando os valores já introduzidos, pela digitação de RVY (Reset Variables Yes, "reinicializar variáveis"). Introduzimos as informações conhecidas: 1.000 para milhagem, 50 para o custo e 1,75 para o preço. Usamos o valor 1/3 como atrito interno (que o TK!Solver transforma em 0,333333) e 0,000095 para a resistência do ar. Pressionando [!] para que o programa execute

Tela 4: Modelo incompleto



o cálculo, surgem então valores para milhagem e combustível *(tela 4)*.

Observa-se que não foram gerados valores para velocidade, tempo e potência — e a velocidade é o valor particularmente necessário. Se o usuário passar da planilha Variáveis para a planilha Regras, verá que, de cinco equações, três permanecem não satisfeitas, como indicado *(tela 5)* por um asterisco na coluna S (status).

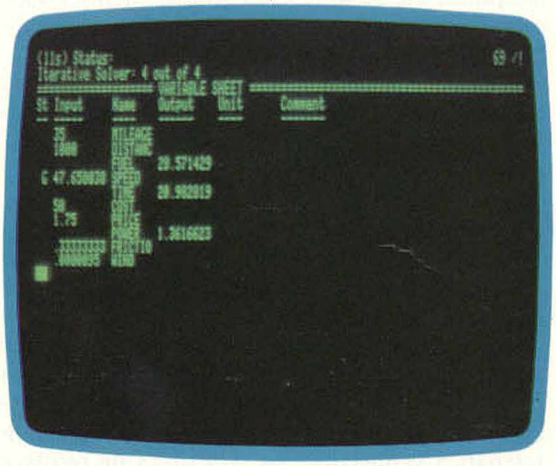
Tela 5: Equações não satisfeitas

Resolução iterativa

Já que não se pode resolver o modelo pela solução direta (Direct Solver), deve-se tentar a iterativa (Iterative Solver). Esta toma um valor inicial e o introduz como tentativa na equação. Se o valor não for correto, o TK!Solver usará uma série de aproximações sucessivas (como no jogo maior-ou-menor), até localizar o valor exato.

A primeira providência é tomar o valor de milhagem gerado anteriormente e colocá-lo na coluna Input. Fazemos isso digitando [I] na coluna St antes da milhagem, na planilha Variáveis. Então estimamos um valor para a velocidade — 50, por exemplo — e o introduzimos na coluna Input. Digitamos G (guess, tentativa) na coluna St e [!] para que se efetuem os cálculos. O TK!Solver apresenta as palavras Iterative Solver na parte superior da tela, e conta cada aproximação. Na quarta tentativa, nesse caso, o programa chega aos valores corretos de velocidade, tempo e potência — indicados na *tela 6*.

Tela 6: Valores usados nas iterações



De acordo com o TK!Solver, é necessária uma velocidade média de pouco mais de 47 milhas por hora para se completar a viagem dentro do orçamento.

GLOSSÁRIO DAS TELAS

MILEAGE = milhagem
DISTANCE = distância
FUEL = combustível
SPEED = velocidade
TIME = tempo
COST = custo
PRICE = preço
POWER = potência
FRICTION = atrito
WIND = vento (resistência do ar)
VARIABLE SHEET = planilha Variáveis
RULE SHEET = planilha Regras

INTRODUÇÃO AO ASSEMBLY

A programação em código de máquina é a chave para se obter o máximo rendimento do microprocessador. Esta primeira parte de nosso curso levará a uma ampla compreensão dos fundamentos da microprogramação.

O código de máquina é uma linguagem de programação e tem este aspecto:

```
INSTK: SBC $D9FA,X;Outport flag value
```

ou:

```
DE23 FD FA D9
```

ou, ainda:

```
11011110 00100011 11111101 11111010 11011001
```

Outras vezes, pode ser encontrado assim:

```
1240 LET ACC = ACC—FLAG (X)
```

e também:

```
PERFORM FLAG-ADJUST THROUGH LOOP 1
```

Todas essas expressões não passam de diferentes versões da mesma instrução em código e, como ele será decifrado por uma máquina de computação, recebe em todos os casos o nome de "código de máquina". Na realidade, para a máquina, ele se apresenta simplesmente como uma configuração de níveis de voltagem ou de corrente elétrica.

Quando nos referimos ao código de máquina, em geral pensamos na linguagem ASSEMBLY, cujo exemplo é a primeira instrução da lista — para o microprocessador 6502; as outras constituem diferentes formas de representar a mesma seqüência de eventos elétricos.

Portanto, o código de máquina (ou linguagem ASSEMBLY — não vamos nos preocupar com a distinção, por enquanto) é apenas mais uma linguagem de programação. Uma vez que a programação sempre vem antes da linguagem — tanto nos programas em ASSEMBLY do IBM, em BASIC do Atari, como numa hipotética "linguagem telepática venusiana" —, você terá de resolver o problema da programação em sua cabeça antes de mexer no teclado. E o modo como expressará sua solução — o algoritmo — determinará a forma final do programa.

Na verdade, trata-se sempre de escolher, entre várias linguagens, aquela que facilitará o desenvolvimento ou a leitura do programa. A solução, contudo, deve vir antes: o conteúdo precede a forma.

Nesse caso, por que chamá-la de código de máquina e por que se preocupar em usá-la? Damos esse nome à linguagem porque seu conjunto de instruções corresponde com exatidão ao conjunto de operações fundamentais, ou "primitivas", que um determinado microprocessador executa. Por meio do código de máquina, controlamos passo a passo a operação do microprocessador, em vez de deixar que um programa interpretador o faça de modo mais abrangente.

O motivo mais comum para o emprego do código de máquina é a velocidade: ao endereçar diretamente o processador, o programa não precisa ser traduzido ou interpretado. Com a eliminação desse estágio intermediário, reduz-se bastante o tempo de execução do programa. No entanto, o processo de codificação, teste, depuração, modificação e manutenção de um código de máquina exigirá, com certeza, pelo menos duas vezes mais tempo do que levaria num programa em linguagem de alto nível.

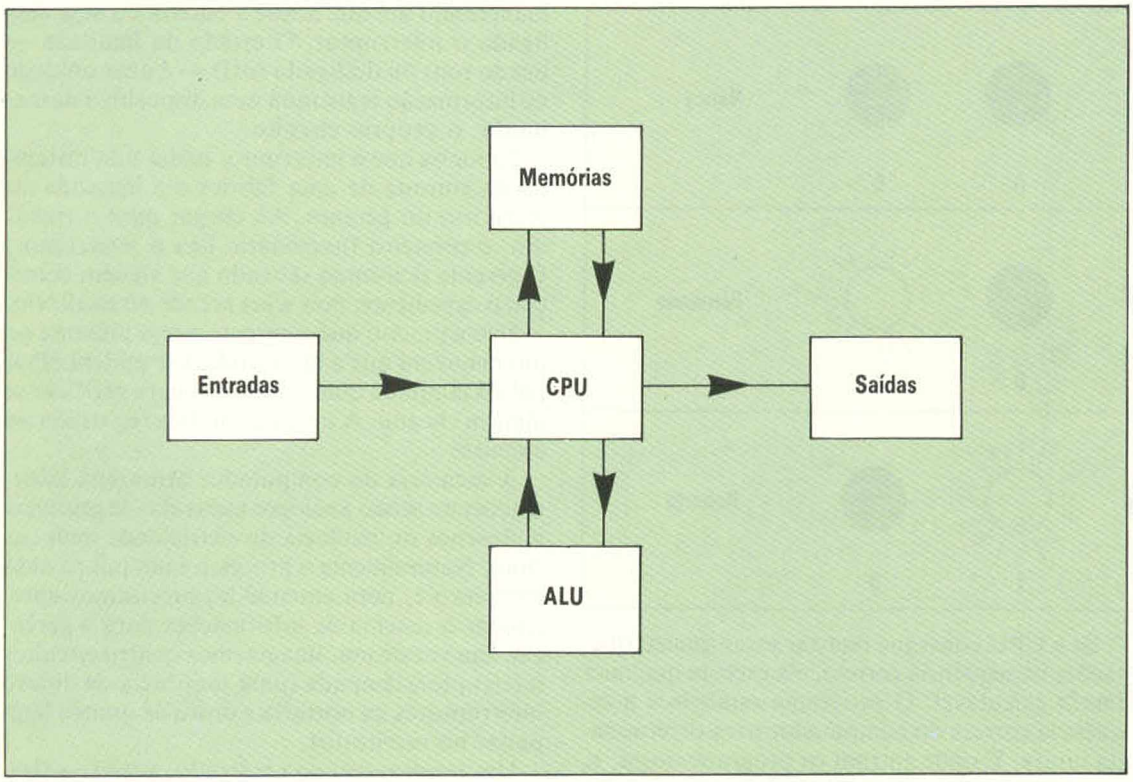
A falta de interação com o programador e a dificuldade de se lidar com o código de máquina foi o principal estímulo para a criação das linguagens de alto nível, como COBOL e BASIC. Vimos que o conjunto de instruções em código de máquina equivale ao conjunto de operações do processador. Mas o que são essas operações e o que faz o processador?

O centro do sistema

Em termos simples, a CPU (Central Processing Unit, "unidade central de processamento") do computador não passa de um interruptor que controla o fluxo de corrente num sistema de computação. Compõem esse sistema a memória, a ALU (Arithmetic and Logic Unit, "unidade aritmética e lógica") e os dispositivos de entrada e saída.

Ao acionar uma tecla, você introduz alguma informação na máquina por meio de uma configuração de voltagem gerada na unidade de teclado. A CPU transfere essa configuração de voltagem para uma posição da memória. Em seguida, transfere uma configuração correspondente, proveniente de algum outro lugar da memória, para a tela, de modo a gerar um determinado padrão de caracteres.

Esse processo é semelhante ao funcionamento de uma máquina de escrever, mas com a diferença de que nesta há uma conexão mecânica entre o acionamento de uma tecla e a impressão do caractere, enquanto num computador essa ligação ocorre porque a CPU transfere configurações corretas de voltagem de um lugar para outro.



Controle global
A CPU determina o fluxo de correntes elétricas — e, portanto, de informações - pelos diversos componentes do sistema. Isso permite a realização das quatro operações (aritméticas, lógicas, de memória e de controle) nas quais se baseia o funcionamento de qualquer computador.

Nem sempre o acionamento de uma tecla faz aparecer um caractere no vídeo: pode também destruir um asteróide, gravar um programa, apagar um arquivo em disco, ou imprimir uma carta. O resultado da operação depende do modo e da finalidade com que a CPU transfere a corrente elétrica.

Modo de funcionamento da CPU

Os procedimentos executados pela CPU classificam-se, para nossos objetivos, nas operações: aritméticas, lógicas, de memória e de controle. Todas resultam de transferências de informações através de diferentes trajetos, no sistema e na CPU, ou seja, para esta todas as operações se assemelham.

Operações aritméticas. A adição e a subtração constituem a característica mais importante da máquina. Ela subtrai por meio da representação negativa de um dos números e sua subseqüente adição com o outro. Por exemplo, 7 + 5 = 12 significa:

(mais 7) somado a (mais 5) igual a (mais 12);

enquanto 7 – 5 = 2 equivale a:

(mais 7) somado a (menos 5) igual a (mais 2).

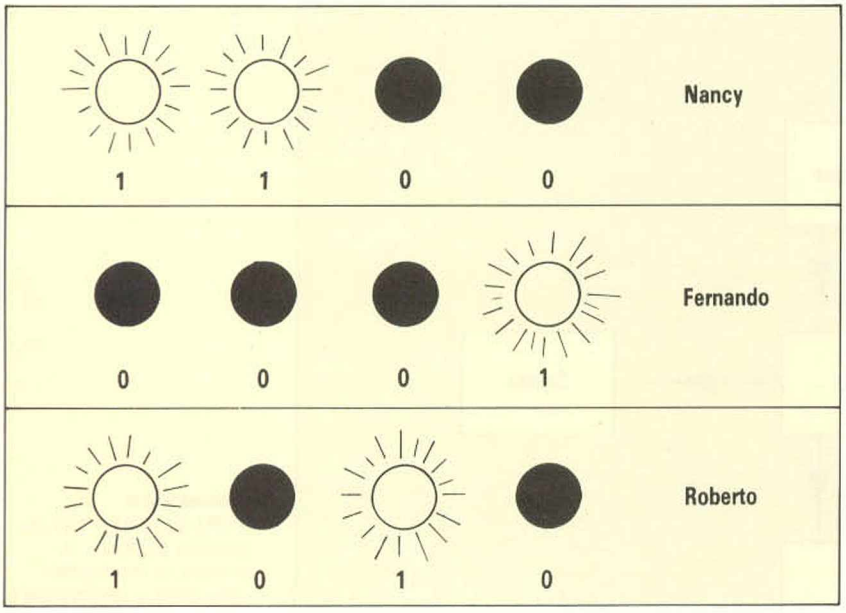
A multiplicação e a divisão são consideradas adições ou subtrações repetidas, de modo que também é possível programar a CPU para simular essas operações. Se a CPU consegue realizar as quatro operações aritméticas, então pode efetuar qualquer outro cálculo matemático. No entanto, lembre-se de que todo seu potencial matemático depende simplesmente da capacidade de somar dois números.

Operações lógicas. Efetuam a comparação de dois números: não apenas em termos de quantidades relativas, mas também em termos da configuração de seus dígitos. É fácil ver que sete é maior que cinco porque extraímos cinco de sete e obtemos um resultado positivo. Além de fazer esse tipo de comparação, a CPU também verifica que, por exemplo, 189 e 102 têm o mesmo dígito na coluna das centenas. Talvez isso não pareça algo extraordinário, mas sua utilidade se tornará mais evidente no decorrer do curso.

Operações de memória. Envolvem tanto a cópia de informações de uma posição qualquer da memória externa para sua própria memória (registro), como de seu registro para uma outra posição da RAM. Executando essas operações em seqüência, a CPU transfere informações de uma parte qualquer da memória para outra. Para que a memória do computador tenha alguma utilidade, é absolutamente necessário que a CPU seja capaz de realizar essas duas operações. Só assim torna-se possível um controle completo da memória.

Operações de controle. Consistem, na verdade, em decisões quanto à seqüência pela qual a CPU executa as outras operações descritas. Por enquanto, tudo o que precisamos saber sobre as operações de controle é que a CPU pode tomar determinadas decisões a respeito de sua própria atividade.

Portanto, a CPU executa operações aritméticas, compara números, desloca informações na memória e decide sobre sua própria seqüência de operações. Essa lista de procedimentos é suficiente para definir uma máquina de computação ideal.



Memória elétrica
Um circuito elétrico com interruptores e lâmpadas serve de analogia ao funcionamento da memória nos computadores. Neste exemplo, o gerente de uma fábrica controla a chegada dos funcionários sem sair de seu escritório, apenas associando cada seqüência de luzes acesas e apagadas aos nomes dos empregados.

Se a CPU consegue realizar essas quatro operações na seqüência correta, ela executa qualquer tarefa calculável. O programa estabelece a seqüência correta do computador para determinada tarefa. É onde entram os programadores; e, se a CPU tivesse a capacidade de gerar suas próprias seqüências de operações, eles não seriam necessários.

Para tornar isso mais claro, consideremos o seguinte: todos os programas incluem variáveis, isto é, os nomes das posições da memória em que são armazenadas as informações. A maioria dos programas executa algum tipo de operação aritmética com essas variáveis.

Após tal operação, o programa muitas vezes compara duas unidades de informação e, como resultado, executa um conjunto ou outro de instruções. Em geral, as informações são introduzidas num programa por meio do teclado, e apresentadas para o usuário no vídeo.

Com exceção da frase sobre dispositivos de entrada e saída, essa descrição não contém nada além das quatro operações elementares apresentadas em termos diferentes. Supondo (ao menos por enquanto) que, para a CPU, todos os dispositivos de entrada/saída sejam áreas especiais da memória, então a imagem do computador ideal está completa.

A memória do circuito

Imagine um circuito elétrico simples, composto de uma bateria, um interruptor e uma lâmpada: se ligamos o interruptor, a luz acende, assim permanecendo até que acabe a bateria ou seja desligado o interruptor. O estado da lâmpada — ligado (on) ou desligado (off) — é uma unidade de informação registrada num dispositivo de memória: o próprio circuito.

Suponha que o interruptor tenha sido instalado na entrada de uma fábrica e a lâmpada no escritório do gerente. Ao chegar para o trabalho, o primeiro funcionário liga o interruptor. O gerente fica então sabendo que alguém começou o expediente, pois a luz acende no escritório.

Não é preciso que o gerente esteja presente no momento em que a luz acende. Ele poderá olhar para a lâmpada quando quiser, para verificar se alguém chegou. A informação fica registrada no circuito.

A memória do computador armazena informações de modo análogo: todas elas se resumem à presença ou ausência de eletricidade num circuito. Naturalmente o processo é um pouco mais complexo e, para entendê-lo, precisamos aperfeiçoar o sistema de informações para a gerência. Em vez de um, imaginemos quatro circuitos interruptor/lâmpada (uma seqüência de quatro interruptores na portaria e outra de quatro lâmpadas no escritório).

Um interruptor, ao ser ligado, acende a lâmpada correspondente. Cada empregado, então, aciona os interruptores numa combinação específica: ao chegar, Nancy liga o primeiro e o segundo interruptores, deixando desligados o terceiro e o quarto; Fernando aciona apenas o quarto; Roberto, o primeiro e o terceiro; e assim por diante. As luzes no escritório indicam ao gerente não apenas que alguém chegou para trabalhar, mas também informam quem.

Suponhamos agora que a posição "desligado" esteja associada ao 0, e a posição "ligado" ao 1. Assim, Nancy coloca os interruptores em 1100 (os dois primeiros ligados, o terceiro e o quarto desligados); Fernando forma a combinação 0001 (só o quarto interruptor ligado).

Se o gerente também associar 1 às lâmpadas acesas e 0 às desligadas, tanto ele quanto os funcionários estarão usando a mesma linguagem de identificação. Para ambos 0001 significa Fernando.

Quantas combinações são possíveis no caso? Cada interruptor pode estar em uma de duas posições e há quatro interruptores. Serão, portanto, 2x2x2x2 = 16 combinações diferentes. Examinemos todas as possibilidades:

```
0000  0001  0010  0011
0100  0101  0110  0111
1000  1001  1010  1011
1100  1101  1110  1111
```

Por mais que tente, você não conseguirá formar mais que dezesseis combinações.

Observe com que rapidez passamos da imagem concreta de lâmpadas numa sala para a questão abstrata das combinações de 1 e 0. Abstraindo um pouco mais, poderemos transformar essas combinações em números.

Equivalência
A cada funcionário desta hipotética empresa está associada uma configuração específica de lâmpadas acesas e apagadas, que pode ser transcrita de modo imediato em números binários. Estes, por sua vez, podem ser convertidos ao sistema decimal.

Nome	n.° decimal	n.° binário
Nancy	12	1100
Fernando	1	0001
Roberto	10	1010

Sistemas de numeração

Pense no processo que empregamos para contar e escrever simultaneamente. Você consegue contar de zero a nove com facilidade porque cada um desses números possui nome e símbolo específicos para representá-lo. Mas o que você escreve depois de nove? Existe um nome (dez) para esse número, mas não um único símbolo para sua representação.

Você é obrigado a reutilizar alguns dos outros símbolos: 10, 11, 12 e assim por diante, até 99, quando novamente se esgotam as combinações possíveis com duas colunas, de modo que o número seguinte deverá ter três colunas (100).

Se isso lhe parece óbvio demais, lembre-se de como foi difícil aprender a contar na escola: todo aquele papel quadriculado e as casas das centenas, dezenas e unidades escritas sobre cada soma! Hoje você sabe que o número 152 significa "1 nas centenas, 5 nas dezenas, 2 nas unidades", ou que 100 + 50 + 2 = 152. A contagem funciona dessa forma porque usamos apenas dez dígitos (0, 1, 2, 3, ..., 9) para representar todos os números possíveis.

Como funcionará então a contagem se tivermos apenas dois dígitos (0 e 1)? Contar até 1 não tem mistério; mas como representaremos o número seguinte? Não dispondo de outros dígitos teremos, portanto, de reutilizar os dois existentes (tal como fizemos ao contar com dez dígitos); e escrevemos o número seguinte como 10.

Sabemos que o número seguinte no sistema decimal chama-se "dois"; portanto, no sistema binário, 10 representa o número 2. O número seguinte na contagem é 3, que representaremos por 11. E depois, o que acontece? Esgotadas as combinações de dois dígitos, representamos o número 4 como 100; o 5 como 101, o 6 como 110 e o 7 como 111. Desse modo, continuamos a contar em números decimais (0, 1, 2 etc.), só que estamos escrevendo em números binários (0, 1, 10, 11, 100, 101, ...).

Da mesma forma que o número decimal 152 equivale a (1x100) + (5x10) + (2x1), o número binário 101 significa (1x4) + (0x2) + (1x1). Em vez de termos as colunas das centenas, dezenas e unidades para esses números, usamos as colunas dos 4, dos 2 e dos 1. Num número decimal, o valor de um dígito multiplica-se por 10 a cada coluna que ele se move para a esquerda; já um número binário é multiplicado por 2.

O sistema binário não passa de uma forma diferente de representar os números. Conhecendo os numerais romanos, você não estranha ao ver um antigo relógio de parede apontar VII horas; e por que não escrever 111 horas? O número efetivo de horas não se altera só porque o representamos de outra maneira.

No entanto, convém chamar o número binário de "um-um-um binário" e escrevê-lo como "111b" para não confundir com uma representação decimal.

```
 1 REM ..................
10 REM . ESTE PROGRAMA ENCHE .
20 REM .   O VIDEO DO TK 85  .
30 REM . COM O CARACTERE ■.  .
40 REM . MANTENHA A LINHA 1. .
50 REM ..................
60 POKE 16514,205
70 POKE 16515,8
80 POKE 16516,8
90 POKE 16517,205
100 POKE 16518,130
110 POKE 16519,64
120 RAND USR 16514
```

```
 1 REM ##########################
10 REM #    ESTE PROGRAMA      #
20 REM #  ILUSTRA A RAPIDEZ    #
30 REM #     DO CODIGO DE      #
40 REM #  MAQUINA NO CONTROLE  #
50 REM #  DO VIDEO DO TK 90X.  #
60 REM #    A LINHA 1 NAO DEVE #
70 REM #     SER SUPRIMIDA OU  #
80 REM #      EDITADA APOS A   #
90 REM #       EXECUCAO DO     #
100 REM #       PROGRAMA.      #
110 REM ##########################
120 LET p=23635: LET endereco=P
EEK (P)+(256*PEEK (P+1))+7
130 BORDER 2
140 DATA 1,0,3,17,0,88,33,0,0,2
37,175,201
150 FOR f=0 TO 11
160 READ codigo
170 POKE endereco+f,codigo
180 NEXT f
190 LET l=0
200 FOR f=0 TO 1 STEP 0
210 POKE endereco+7,l
220 LET d=USR endereco
230 LET l=l+13
240 IF l>=256 THEN LET l=l-256*
INT (l/256)
250 NEXT f
260 STOP
```

Alta velocidade
Estes dois pequenos programas para o TK 85 e o TK 90X, embora escritos em BASIC, ilustram a grande vantagem na programação em código de máquina sobre as linguagens de alto nível: o processamento é muito mais rápido.

PSION

Série integrada
O Xchange, da Psion, é um conjunto de programas integrados com base no software criado para o Sinclair QL. O sistema inclui o processador de texto Quill, o banco de dados Archive, a planilha financeira Abacus e o gerador de gráficos Easel. Os quatro pacotes são comercializados juntos ou em separado. O Xchange foi preparado com versões para os micros IBM PC e PC XT, Apricot e Apricot XI da ACT e Sirius I. Planejam-se versões para o Macintosh da Apple e para o Digital Rainbow.

A Psion consolidou sua imagem produzindo aplicativos para a Sinclair Research. Contudo, numa iniciativa pioneira, surpreende o mercado ao anunciar seu primeiro micro portátil, o Organiser.

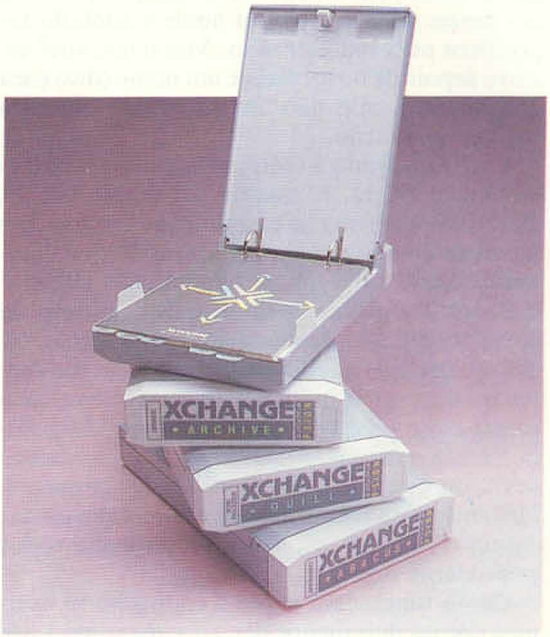
Fundada em 1981 por David Potter, o primeiro lançamento da companhia inglesa Psion foi um conjunto de programas para o então recém-lançado micro ZX81, da Sinclair: Flight Simulator, Backgammon, Vu-Calc e Vu-File. Essa pequena série de softwares, composta por um programa simulador de vôo, um jogo de gamão, uma planilha e um banco de dados, tornou o nome da empresa conhecido da noite para o dia. Por isso, um ano depois, não houve surpresa quando a Sinclair escolheu a Psion para desenvolver um cassete de demonstração para o Spectrum. Graças à enorme popularidade dos equipamentos Sinclair, foi rápida a disseminação dos produtos Psion.

Alguns sucessos da companhia são notáveis, como a venda das versões do Flight Simulator para os equipamentos ZX81 e Spectrum, que ultrapassou as 500.000 cópias. Segundo cálculos de mercado, o total de venda mundial dos cassetes da Psion excedeu os 3 milhões. A partir do lançamento da série Xchange, a Psion deu início a um programa de diversificação de seus produtos. Para isso investiu cerca de 2 milhões de libras, garantindo sua participação no mercado de softwares profissionais.

A Psion sempre se caracterizou por uma abordagem inovadora. No desenvolvimento de seus softwares para micros, procurou usar técnicas avançadas como a "compilação cruzada", que permite a elaboração, em determinado equipamento, de um programa que será rodado em outra máquina. Assim, o Horizons, destinado ao Spectrum, foi desenvolvido num TRS-80.

Atualmente a empresa utiliza dois minicomputadores VAX 750 para criação de seus softwares. Nesses equipamentos, foi desenvolvida a série de programas para o micro QL: a planilha Abacus, o banco de dados Archive, o gerador de gráficos Easel e o processador de textos Quill. Por sua vez, a Psion Support organizou a QLUB, uma linha de apoio aos usuários do QL, respondendo por escrito às dúvidas no prazo de 48 horas.

Estuda-se também a criação de sistemas profissionais para os computadores IBM PC, Apricot, Sirius, DEC Rainbow e Macintosh. A mesma estrutura da série de aplicativos para o



QL repete-se na série Xchange, que se compõe de uma planilha, um banco de dados, um gerador de gráficos e um processador de textos. Esses dois pacotes de software se destacam dos concorrentes pela facilidade com que os dados podem ser transferidos de um para outro.

Um produto da companhia que também chamou muita atenção foi o microcomputador portátil Organiser. Para o relações-públicas da empresa, Robin Kinnear, o fato se explica justamente por se tratar de uma empresa especializada em software: "O Organiser é um projeto muito atraente em termos de compactação de software, e não havia nada parecido no mercado". Assim, a Psion decidiu fabricar seu próprio hardware, influenciada pela grande experiência com software.

Para o Organiser são produzidos três aplicativos — de ciências, matemática e finanças —, juntamente com módulos de RAM de 8 Kbytes e de 16 Kbytes, os chamados "datapacks".

A Psion avalia também a possibilidade de acrescentar outros pacotes de programas e, nesse sentido, vários projetistas se mostraram interessados em desenvolver software para o Organiser.

A outra prioridade da Psion se refere a sua expansão no mercado internacional. A empresa mantém subsidiárias nos Estados Unidos e na África do Sul, pretendendo também garantir a presença de seus produtos na Europa mediante contratos com distribuidoras locais. Além disso, a ligação com a Sinclair contribui para os planos da Psion de conquistar o mercado da Europa oriental. O primeiro passo nessa direção foi dado pela exportação de quatrocentas unidades do ZX81 para a Tchecoslováquia.

Da teoria à prática
A Psion tem como fundador e sócio majoritário David Potter, especialista em Física Computacional pelo Imperial College, de Londres, e pela Universidade da Califórnia. A empresa chamava-se antes Potter Scientific Investments.

```
126 CO=CO*.9999:NEXT
127 IF RF<>0 THEN XDRAW 1 AT XP,YP:GOSUB
    165:XDRAW 1 AT XP,YP
128 GOTO 115
129 XDRAW 1 AT XP,YP:VTAB 21:HTAB 2:PRINT
    "OUTRO JOGO ? (S/N) ";
130 GET A$:IF A$="N" OR A$="n" THEN TEXT
    :HOME :END
131 IF A$="S" OR A$="s" THEN 111
132 GOTO 130
133 HPLOT XT,YT TO 0,0:HPLOT XT,YP TO 0
    ,159
134 HPLOT XT,YT TO 279,0:HPLOT XT,YT TO
    279,159
135 RETURN
136 REM
137 REM * LE TECLADO E DEFINE *
138 REM * NOVO MOVIMENTO.      *
139 REM
140 I=0:A=PEEK(49152)
141 IF A>127 THEN POKE 49168,0:A$=CHR$(
    A-128):FOR I=1 TO 6:IF A$<>MID$("JI
    LK P",I,1) THEN NEXT
142 RETURN
143 GOSUB 189:XP=XP-(XP>6)*7:GOTO 190
144 GOSUB 189:YP=YP-(YP>7)*8:GOTO 190
145 GOSUB 189:XP=XP+(XP<273)*7:GOTO 190
146 GOSUB 189:YP=YP+(YP<152)*8:GOTO 190
147 HCOLOR=7:XDRAW 1 AT XP,YP:HPLOT 0,1
    59 TO XP,YP:HPLOT 279,159 TO XP,YP
148 HCOLOR=0:HPLOT 0,159 TO XP,YP:HPLOT
    279,159 TO XP,YP
149 XC=INT(XP/7)+1:YC=INT(YP/8)+1:FOR J
    =0 TO 3:IF PZ(J)=1000 THEN 154
150 IF YC<>INT(PY(J)) THEN 154
151 IF XC<INT(PX(J))-(PZ(J)<=15)-(PZ(J)
    <=30) THEN 154
152 IF XC>INT(PX(J))+(PZ(J)<=15) THEN 1
    54
153 PZ(J)=-PZ(J):GOSUB 165:SC=SC+100:
    GOSUB 191:GOTO 155
154 NEXT
155 XDRAW 1 AT XP,YP:SH=SH-20:GOSUB 196
    :RETURN
156 GET A$:RETURN
157 IF RND(1)<CO THEN 160
158 PX(M)=RND(1)*35+3:PY(M)=RND(1)*20+1
    :PZ(M)=45
159 R=RND(1)-.5:VX(M)=(ABS(R)-.25)*VY(M
    )=SQR(.0625-VX(M)*VX(M))*SGN(R):RF=
    1
160 RETURN
161 PX(M)=PX(M)+VX(M)*(PX(M)>4 AND PX(M
    )<37):IF INT(QX(M))<>INT(PX(M)) THEN
    RF=1
162 PY(M)=PY(M)+VY(M)*(PY(M)>2 AND PY(M
    )<20):IF INT(QY(M))<>INT(PY(M)) THEN
    RF=1
163 PZ(M)=PZ(M)-2*(PZ(M)>2):IF PZ(M)=30
     OR PZ(M)=15 THEN RF=1
164 RETURN
165 FOR I=0 TO 3:IF QZ(I)=1000 THEN 173
166 NF=QZ(I):QZ(I)=ABS(QZ(I))
167 IF QZ(I)<=15 THEN GOSUB 180:GOTO 17
    0
168 IF QZ(I)<=30 THEN GOSUB 181:GOTO 17
    0
169 GOSUB 182
170 IF NF>=0 THEN 173
171 IF I<3 THEN GOSUB 187:I=I-1
172 QZ(3)=1000
173 NEXT :FOR I=3 TO 0 STEP -1:IF PZ(I)
    =1000 THEN 178
174 QX(I)=PX(I):QY(I)=PY(I):QZ(I)=PZ(I)
175 IF ABS(PZ(I))<=15 THEN GOSUB 183:GOTO
    178
176 IF ABS(PZ(I))<=30 THEN GOSUB 184:GOTO
    178
177 GOSUB 185
178 NEXT
179 RETURN
180 HTAB QX(I)-2:VTAB QY(I):PRINT "
    ":RETURN
181 HTAB QX(I)-1:VTAB QY(I):PRINT "  ":
    RETURN
182 HTAB QX(I):VTAB QY(I):PRINT " ":RETURN
183 GOSUB 186:HTAB PX(I)-2:VTAB PY(I):PRINT
    P$(PH):RETURN
184 GOSUB 186:HTAB PX(I)-1:VTAB PY(I):PRINT
    P$(PH+3):RETURN
185 GOSUB 186:HTAB PX(I):VTAB PY(I):PRINT
    P$(PH+6):RETURN
186 PH=(PZ(I)>=0)*INT(PX(I)-2*INT(PX(I)
    /2)+1):RETURN
187 FOR K=I TO 2:PX(K)=PX(K+1):PY(K)=PY
    (K+1):PZ(K)=PZ(K+1)
188 VX(K)=VX(K+1):VY(K)=VY(K+1):QX(K)=Q
    X(K+1):QY(K)=QY(K+1):QZ(K)=QZ(K+1):
    NEXT :PZ(3)=1000:RETURN
189 OX=XP:OY=YP:RETURN
190 XDRAW 1 AT OX,OY:XDRAW 1 AT XP,YP:RETURN
191 N$=STR$(SC):VTAB 22:HTAB 11:GOSUB 1
    98:R=INT(SC/2000):IF R>4 THEN R=4
192 VTAB 24:HTAB 16:CALL -868:PRINT R$(
    R);:RETURN
193 REM
194 REM * PREPARA TELA          *
195 REM
196 IF SH<0 THEN SH=0
197 N$=STR$(SH):VTAB 22:HTAB 31:GOTO 19
    8
198 IF LEN(N$)<5 THEN PRINT LEFT$("0000
    ",5-LEN(N$));
199 PRINT N$;:RETURN
200 HOME :HGR :INVERSE :VTAB 22:HTAB 2:
    PRINT " PONTOS ";:HTAB 18:PRINT " P
    OTENCIA ";
201 VTAB 24:HTAB 02:PRINT " GRADUACAO "
    ;
202 NORMAL :RETURN
203 POKE 232,100:POKE 233,3
204 POKE 868,1:POKE 870,4:POKE 871,0
205 FOR I=0 TO 4:READ A:POKE 872+I,A:NEXT
206 HCOLOR=7:ROT=0:SCALE=4
207 FOR I=0 TO 4:READ R$(I):NEXT
208 FOR I=768 TO I+87:READ A:POKE I,A:NEXT
209 FOR I=138*256 TO I+175:READ A:POKE
    I,A:NEXT
210 IF PEEK(191*256)=76 THEN PRINT D$;"
    PR#A$300":GOTO 212
211 POKE 54,0:POKE 55,3:CALL 1002
212 POKE 6,0:POKE 7,138:RETURN
213 DATA 176,12,31,5,0
214 REM
215 REM *          GRADUACOES          *
216 REM
217 DATA SARGENTO,CAPITAO,MAJOR,CORONEL
    ,GENERAL
218 REM
219 REM * SUB-ROTINA EM CODIGO DE *
220 REM *          MAQUINA          *
221 REM
222 DATA 216,120,133,69,134,70,132,71,1
    66,7,10,10,176,4,16,62,48,4,16,1,23
    2,232,10,134,27,24,101,6,133,26
223 DATA 144,2,230,27,165,40,133,8,165,
    41,41,3,5,230,133,9,162,8,160,0,177
    ,26,36,50,48,2,73,127,164,36
224 DATA  145,8,230,26,208,2,230,27,165
    ,9,24,105,4,133,9,202,208,226,165,6
    9,166,70,164,71,88,76,240,253
225 REM
226 REM * RE-DEFINICAO DE CARACTE- *
227 REM * RES. (NAVES INIMIGAS)     *
228 REM
229 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,40,4
    2,2,64,64,96,16,21,117,112,0,0,0,1,
    2,42,43,3,0,0,0,0,0,0,5,21,16
230 DATA  0,0,0,0,64,104,66,0,0,0,0,1,3
    ,23,67,0,0,0,0,0,8,42,0,0,0,0,0,0,6
    4,84,21,1,32,32,112,8,10,122,120,0
231 DATA 0,0,0,1,21,85,65,0,0,0,0,0,0,2
    ,10,8,0,0,0,64,96,116,97,0,0,0,0,0,
    1,11,33,0,0,0,0,0,4,21,0,0
232 DATA 24,48,24,64,118,3,56,0,56,99,4
    8,55,88,111,102,0,6,12,63,27,113,31
    ,112,0,3,6,48,108,12,51,0,0,0,56,99
    ,48,55,88,0,0
233 DATA 0,6,12,63,27,113,0,0,0,0,76,11
    8,54,0,0,0
234 REM
235 REM *    INSTRUCOES          *
236 REM
237 TEXT :HOME :VTAB 12:HTAB 05:PRINT "
    DESEJA INSTRUCOES ? (S/N) ";
238 GET A$:IF A$="S" OR A$="s" THEN 241
239 IF A$="N" OR A$="n" THEN RETURN
240 GOTO 238
241 HOME :HTAB 10:INVERSE :PRINT "  GUE
    RRA GALACTICA  ":NORMAL
242 PRINT :PRINT "VOCE ESTA  EM UM CRUZ
    ADOR  GALACTICO EM"
243 PRINT :PRINT "DIRECAO AO  SISTEMA D
    E ANDROMEDA.  SUA "
244 PRINT :PRINT "MISSAO E' A DE LEVAR
    UMA CARGA PRECIOSA"
245 PRINT :PRINT "A SALVO DE PIRATAS DO
     ESPACO QUE TENTA-"
246 PRINT :PRINT "RAO INTERCEPTA-LO. SU
    A  NAVE ESTA EQUI-"
247 PRINT :PRINT "PADA  COM UMA PODEROS
    A  ARMA  LAZER QUE"
248 PRINT :PRINT "CONSUMIRA 20 PONTOS D
    E SUA POTENCIA CA-"
249 PRINT :PRINT "DA VEZ  QUE ACIONADA.
     POREM 100 PONTOS"
250 PRINT :PRINT "SERAO  PERDIDOS CADA
    VEZ QUE O INIMIGO "
251 PRINT :PRINT "ACERTE SUA NAVE."
252 PRINT :PRINT "QUALQUER TECLA PARA C
    ONTINUAR ";:GET A$
253 HOME :PRINT "SUA  PROMOCAO D
    EPENDERA  DE SUA PERFOR-"
254 PRINT :PRINT "MANCE PERANTE O INIMI
    GO.  A CADA 20 NA-"
255 PRINT :PRINT "VES INIMIGAS ABATIDAS
    ,  VOCE SUBIRA UM "
256 PRINT :PRINT "POSTO."
257 PRINT :PRINT "INTRUCOES SOBRE PILOT
    AGEM:"
258 PRINT :PRINT "                  CIMA"
259 PRINT :PRINT "                    I"
260 PRINT :PRINT "  ESQUERDA      J    L
     DIREITA"
261 PRINT :PRINT "                  K"
262 PRINT :PRINT "                BAIXO"
263 PRINT :PRINT "USE A BARRA DE ESPACO
     PARA ATIRAR"
264 PRINT :PRINT "QUALQUER TECLA PARA C
    ONTINUAR. ";:GET A$
265 RETURN
```


EXPLOSÃO

PARA LINHA APPLE

Aqui a disputa é contra o computador. Antes de iniciar, escolha o tabuleiro: pode ser 3 * 3 ou 4 * 4. Você coloca suas peças numa casa e o computador em outra, alternadamente. Cada posição do tabuleiro comporta apenas determinado número de peças. Ultrapassado esse limite, ocorre uma "explosão", e a peça a mais é lançada às posições vizinhas. Estas, por sua vez, se também estiverem no limite, explodirão, espalhando peças e provocando uma reação em cadeia. Ganha quem conseguir gerar uma explosão sem fim.

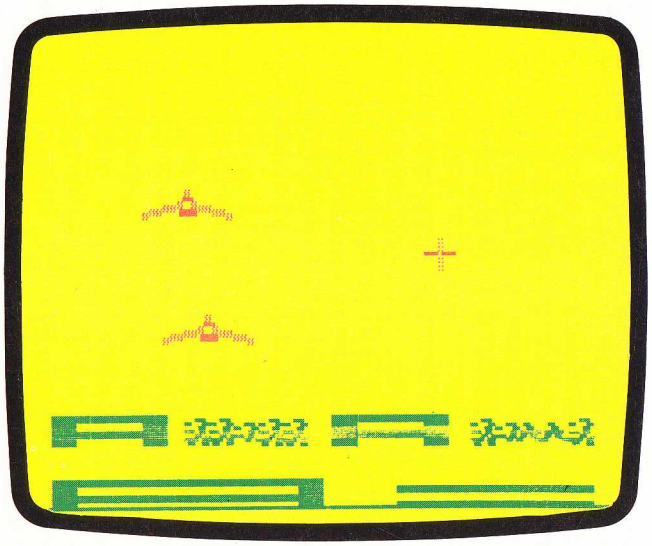
```
1000 REM*********************************
1010 REM*        PARA LINHA APPLE       *
1020 REM*                               *
1030 REM*            EXPLOSAO           *
1040 REM*********************************
1050 GOSUB 2410
1060 DIM RB(5,5),SB(5,5),ST(5,5)
1070 C1$="I---I---I---I---I"
1080 C2$="I   I   I   I   I"
1090 BE$=CHR$(7):ER$=BE$+BE$+BE$:ER$=ER$
     +ER$
1100 GOTO 2320
1110 FOR X=1 TO SI:FOR Y=1 TO SI:SB(X,Y)
     =RB(X,Y):NEXT Y,X:RETURN
1120 FOR X=1 TO SI:FOR Y=1 TO SI:RB(X,Y)
     =SB(X,Y):NEXT Y,X:RETURN
1130 REM
1140 REM  ** INICIALIZA **
1150 HOME :INPUT "TAMANHO DO TABULEIRO (
     3 OU 4): ";SI
1160 IF SI<3 OR SI>4 THEN 1150
1170 FOR X=1 TO SI
1180 FOR Y=1 TO SI
1190 ST(X,Y)=4-(X=1)-(X=SI)-(Y=1)-(Y=SI)
1200 NEXT Y,X
1210 PRINT :INPUT "QUEM COMECA ? VOCE(V)
     OU EU(E): ";AN$
1220 CM=(LEFT$(AN$,1)="E")
1230 REM  ** DESENHA TABULEIRO **
1240 HOME
1250 HTAB 12:PRINT LEFT$("\ X 1   2   3
     4",13+4*(SI=4))
1260 HTAB 13:PRINT "\":HTAB 12:PRINT "Y"
     ;
1270 FOR I=1 TO SI
1280 HTAB 14:PRINT LEFT$(C1$,13+4*(SI=4)
     )
1290 HTAB 14:PRINT LEFT$(C2$,13+4*(SI=4)
     )
1300 HTAB 12:PRINT I;" "+LEFT$(C2$,13+4*
     (SI=4))
1310 HTAB 14:PRINT LEFT$(C2$,13+4*(SI=4)
     )
1320 NEXT I
1330 HTAB 14:PRINT LEFT$(C1$,13+4*(SI=4)
     )
1340 RETURN
1350 REM
```

```
1360 REM  ** EXECUTA MOVIMENTO **
1370 VTAB 21:HTAB 1:CALL -958
1380 HTAB 16:PRINT "SUA VEZ"
1390 HTAB 19:INPUT "X: ";X$
1400 MX=VAL(X$):IF MX<1 OR MX>SI THEN 13
     70
1410 HTAB 19:INPUT "Y: ";Y$
1420 MY=VAL(Y$):IF MY<1 OR MY>SI THEN 13
     70
1430 IF RB(MX,MY)<0 THEN PRINT ER$;:GOTO
     1370
1440 GOSUB 1110
1450 X=MX:Y=MY:DI=1
1460 GOSUB 1510
1470 IF ET THEN 2240
1480 GOSUB 1120
1490 RETURN
1500 REM
1510 REM   ** EXPLOSAO ETERNA **
1520 SB(X,Y)=SB(X,Y)+PL
1530 NE=0
1540 IF PL=-1 THEN INVERSE
1550 IF DI THEN VTAB 1+4*Y:HTAB 12+4*X:
     PRINT ABS(SB(X,Y));
1560 NORMAL
1570 XP=0
1580 FOR X=1 TO SI
1590 FOR Y=1 TO SI
1600 IF ABS(SB(X,Y))<ST(X,Y) THEN 1690
1610 XP=1
1620 NE=NE+1
1630 SB(X,Y)=0
1640 IF DI THEN VTAB 1+4*Y:HTAB 12+4*X:
     PRINT " ";
1650 EX=X:EY=Y+1:GOSUB 1750
1660 EX=X+1:EY=Y:GOSUB 1750
1670 EX=X:EY=Y-1:GOSUB 1750
1680 EX=X-1:EY=Y:GOSUB 1750
1690 NEXT Y,X
1700 ET=(NE>SI*SI)
1710 IF XP AND NOT ET THEN 1570
1720 IF (DI AND ET) AND (NE<3*SI*SI) THEN
     1580
1730 RETURN
1740 REM
1750 REM  ** SOMA POSICOES VIZINHAS **
1760 SB(EX,EY)=PL*(ABS(SB(EX,EY))+1)
1770 IF NOT DI OR ST(EX,EY)=0 THEN 1820
1780 IF PL=-1 THEN INVERSE
1790 VTAB 1+4*EY:HTAB 12+4*EX:PRINT ABS(
     SB(EX,EY));
1800 FOR DL=1 TO 300:NEXT
1810 NORMAL
1820 RETURN
1830 REM
1840 REM  ** VEZ DO COMPUTADOR **
1850 VTAB 21:HTAB 1:CALL -958
1860 HTAB 17:PRINT "MINHA VEZ"
1870 HTAB 15:PRINT "AGUARDE..."
1880 BE=1000
1890 FOR TX=1 TO SI
1900 FOR TY=1 TO SI
1910 IF RB(TX,TY)>0 THEN 1980
1920 GOSUB 1110
1930 X=TX:Y=TY:DI=0
1940 GOSUB 1510
1950 IF ET THEN MX=TX:MY=TY:GOTO 1990
1960 GOSUB 2100
1970 IF EN<BE OR (EN=BE AND INT(RND(1)*1
     0)<3) THEN BE=EN:MX=TX:MY=TY
1980 NEXT TY,TX
1990 REM  **  MOVIMENTO ATUAL  **
2000 GOSUB 1110
2010 X=MX:Y=MY:DI=1
2020 VTAB 22:HTAB 1:CALL -958
2030 HTAB 19:PRINT "X: "X;BE$
2040 HTAB 19:PRINT "Y: "Y
2050 GOSUB 1510
2060 IF ET THEN 2240
2070 GOSUB 1120
2080 RETURN
2090 REM
2100 REM  ** ANALISA SITUACAO **
2110 EN=0
2120 FOR X=1 TO SI
2130 FOR Y=1 TO SI
2140 EN=EN+SB(X,Y)
2150 IF -SB(X,Y)<ST(X,Y)-1 THEN 2210
2160 EN=EN-2
2170 IF SB(X+1,Y)=ST(X+1,Y)-1 THEN EN=EN
     +10
2180 IF SB(X,Y+1)=ST(X,Y+1)-1 THEN EN=EN
     +10
2190 IF SB(X-1,Y)=ST(X-1,Y)-1 THEN EN=EN
     +10
2200 IF SB(X,Y-1)=ST(X,Y-1)-1 THEN EN=EN
     +10
2210 NEXT Y,X
2220 RETURN
2230 REM
2240 REM  **   FIM   **
2250 VTAB 21:HTAB 1:CALL -958
2260 PRINT "UMA EXPLOSAO SEM FIM. "
2270 IF PL=1 THEN PRINT "VOCE GANHOU ";:
     GOTO 2290
2280 PRINT "EU GANHEI ";
2290 PRINT "EM "NT" MOVIMENTOS ";
2300 END
2310 REM
2320 REM  **  PROGRAMA PRINCIPAL  **
2330 GOSUB 1140:REM INITIALIZE
2340 IF CM THEN 2360
2350 PL=1:GOSUB 1360
2360 PL=-1:GOSUB 1840
2370 :VTAB 24:HTAB 8:PRINT "Q/Q TECLA PA
     RA CONTINUAR";
2380 GET A$
2390 NT=NT+1
2400 GOTO 2350
2410 REM **  INSTRUCOES  **
2420 HOME :HTAB 15:PRINT "EXPLOSAO"
2430 PRINT :PRINT "ESTE E' UM JOGO ENTRE
      VOCE E O COMPUTA-"
2440 PRINT :PRINT "TADOR. UM TABULEIRO 3
     *3 OU 4*4, DE A-"
2450 PRINT :PRINT "CORDO COM SUA ESCOLHA
     , SERA MONTADO E"
2460 PRINT :PRINT "VOCE COLOCARA SUAS PE
     CAS AONDE QUISER"
2470 PRINT :PRINT "INTERCALANDO A VEZ CO
     M O COMPUTADOR."
2480 PRINT :PRINT "CADA POSICAO NO TABUL
     EIRO COMPORTA UM"
2490 PRINT :PRINT "DETERMINADO NUMERO DE
      PECAS, E QUANDO"
2500 PRINT :PRINT "A CAPACIDADE DE UMA P
     OSICAO 'EXPLODIR',"
2510 PRINT :PRINT "AS PECAS CONTIDAS NES
     TA POSICAO SERAO"
2520 PRINT :PRINT "LANCADAS 'AS POSICOES
      AO SEU REDOR,"
2530 PRINT :PRINT "Q/Q TECLA PARA CONTIN
     UAR ";:GET A$
2540 HOME :PRINT "FAZENDO COM QUE AS POS
     ICOES QUE ESTAO"
2550 PRINT :PRINT "COM SUA CAPACIDADE NO
      LIMITE, EXPLODAM"
2560 PRINT :PRINT "TAMBEM, ESPALHANDO SU
     AS PECAS FORMANDO"
2570 PRINT :PRINT "UMA REACAO EM CADEIA.
     "
2580 PRINT :PRINT "GANHA AQUELE QUE CONS
     EGUIR GERAR ESTA"
2590 PRINT :PRINT "EXPLOSAO SEM FIM."
2600 VTAB 22:PRINT "Q/Q TECLA PARA CONTI
     NUAR ";:GET A$
2610 RETURN
```


GUERRA GALÁCTICA

PARA LINHA APPLE

Vamos ver como você se sai nesta perigosa missão: atravessar as fronteiras do espaço, em meio ao jogo de naves inimigas. Seu cruzador espacial está provido de uma poderosa arma laser, com a qual pode destruir os oponentes. Mas aja rápido, pois eles são implacáveis, não hesitando em atacá-lo assim que estiverem em posição de tiro. Sua nave perde cem pontos de potência a cada impacto recebido, e, ao término da carga, você estará eliminado. Este programa foi desenvolvido em BASIC, com uma pequena sub-rotina em linguagem de máquina para plotagem de caracteres em tela de alta resolução. Essa sub-rotina, chamada HROUT, serve para definir as naves inimigas. Trata-se de caracteres comuns. A mira de alvo é produzida por uma tabela de forma. Para controlar a nave pressione: [I], para cima; [J], para a esquerda; [L], para a direita; e [K], para baixo. Use a barra de espaço para atirar.



```
100 REM  ****************************
101 REM  **      PARA LINHA APPLE     **
102 REM  **                           **
103 REM  **     GUERRA GALACTICA      **
104 REM  ****************************
105 REM
106 D$=CHR$(4):DIM P$(8),PX(3),PY(3),PZ
    (3),VX(3),VY(3),QX(3),QY(3),QZ(3),R
    $(4)
107 GOSUB 235
108 GOSUB 203
109 P$(0)="/012":P$(1)="()*+":P$(2)="!"
    +CHR$(34)+"#$":P$(3)="34":P$(4)="%&
    .":P$(5)=",-"
110 P$(6)="5":P$(7)=".":P$(8)="'"
111 GOSUB 200
112 FOR I=0 TO 3:PZ(I)=1000:QZ(I)=1000:
    NEXT
113 CO=.95:SH=5000:SC=0:GOSUB 191:GOSUB
    196
114 XP=52:YP=59:XDRAW 1 AT XP,YP
115 RF=0:FOR M=0 TO 3
116 IF SH=0 THEN 129
117 I=FRE(0):GOSUB 140:ON IGOSUB 143,14
    4,145,146,147,156
118 IF PZ(M)=1000 THEN GOSUB 157:GOTO 1
    27
119 IF PZ(M)<0 THEN RF=1:GOTO 126
120 GOSUB 161
121 IF PZ(M)>15 THEN 126
122 IF RND(1)<CO*.8 THEN 126
123 XDRAW 1 AT XP,YP:XT=INT(PX(M))*7-7:
    YT=INT(PY(M))*8-4
124 HCOLOR=5:GOSUB 133:HCOLOR=0:GOSUB 1
    33:GOSUB 165:XDRAW 1 AT XP,YP
125 SH=SH-100:GOSUB 196
```